ABSTRACT

Title of dissertation: NONLINEAR FLUID–STRUCTURE INTERACTIONS IN FLAPPING WING SYSTEMS

Timothy Fitzgerald,
Doctor of Philosophy, 2013

Dissertation directed by: Professor Balakumar Balachandran
Department of Mechanical Engineering

This work relates to fluid-structure interactions in the context of flapping wing systems. System models of flapping flight are explored by using a coupling scheme to provide communication between a fluid model and a structural model describing a flexible wing. The constructed computational models serve as a tool for investigating complex fluid-structure interactions and characterizing them. Primary goals of this work are construction of models to understand nonlinear phenomena associated with the flexible flapping wing systems, and explore means and methods to enhance their performance characteristics. Several system analysis tools are employed to characterize the coupled fluid-structure system dynamics, including proper orthogonal decomposition, dimension calculations, time histories, and frequency spectra.

Results obtained from two-dimensional simulations conducted for a combination of a two-link structural system and a fluid system are presented and discussed. Comparisons are made between the use of direct numerical simulation and the unsteady vortex lattice method as the fluid model in this coupled dynamical system. To enable three-dimensional studies, a novel solid model is formulated from continuum mechanics for geometrically exact finite elements. A new partitioned fluid-structure

interaction algorithm based on the Generalized-α method is formulated and implemented in a large scale fluids solver inside the FLASH framework. Consistent boundary conditions are also formulated by using Lagrangian particles. Several examples demonstrating the effectiveness of the methods and implementation are shown, in particular, for flapping flight at low Reynolds numbers. Unique experiments have also been undertaken to determine the first few natural frequencies and mode shapes associated with hawkmoth wings. The computational framework developed in this dissertation and the research findings can be used as a basis to understand the role of flexibility in flapping wing systems, further explore the complex dynamics of flapping wing systems, and also develop design schemes that might make use of nonlinear phenomena for performance enhancement.

NONLINEAR FLUID–STRUCTURE INTERACTIONS IN FLAPPING WING SYSTEMS

by

Timothy Fitzgerald

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2013

Advisory Committee:
Professor Balakumar Balachandran, Chair and Advisor
Professor Elias Balaras, George Washington University
Professor Amr Baz
Associate Professor Nikhil Chopra
Professor Sung Lee, Dean's Representative

Dedication

To my dear wife, Stacey.

Acknowledgments

First, I would like to thank Professor Balachandran for his ever present mentorship throughout my time at the University of Maryland. As his student, I learned not only about the technical aspects of research, professional issues of all kinds, but also gained a hunger to explore difficult problems.

I also thank each member of the committee, not only for their guidance on this dissertation, but also for how each has helped me grow and learn during my time at Maryland. Each has been a teacher, a mentor, and a role model that I shall strive to imitate for future students. Thank you also to Dr. Paul Wojciechowski, who started me on the path to engineering and graduate school.

I am deeply indebted to my fellow students and lab mates for their help and support. I am honored to call them my friends. I thank Dr. Marcos Vanella whose longtime friendship has nurtured many fascinating discussions, and for his help in the FLASH implementations. A special thank you is reserved for Marcelo Valdez for his immutable friendship, and his insightful understanding of continuum mechanics.

I also thank my family, from my parents to my wife, for their inexhaustible support. As Newton stood on the shoulders of giants, I stand with the help of my family.

I am thankful for all the benefactors that funded this work. This includes the U.S. Air Force Office of Scientific Research (Grant No. FA95500610093), the U.S. Army Research Office (Grant No. W911NF0610369), a joint venture between the U.S. Air Force Research Laboratory Wright–Patterson Air Force Base, Ohio and the University of Maryland College Park (Cooperative Agreement FA86501023012), and the U.S. National Science Foundation (Grant No. CMMI-1250187). I am also grateful and proud of receiving the Sikorsky Fellowship, and support from the Minta Martin Foundation.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AMR	Adaptive mesh refinement
BLAS	Basic linear algebra subprograms
CFL	Courant-Friedrichs-Lewy
DOF	Degrees of freedom
FEM	Finite element method
FFT	Fast Fourier Transform or discrete Fourier transform
FSI	Fluid-structure interactions
G-α	Generalized-α method
HPC	High performance computing
PDE	Partial differential equation
POD	Proper orthogonal decomposition
RHS	Right hand side of an equation
UVLM	Unsteady vortex lattice method

Chapter 1

Introduction

1.1 Problem of interest

Understanding complicated interactions like those between a fluid and a structure have countless applications both in engineering synthesis as well as biological explanations of nature. In low Reynolds number flows, these scenarios arise in everything from the mimicking of swimming fish to explaining the mechanics of dragonfly agility. Flapping motion is a common method of locomotion exhibited by many natural systems and serves as the inspiration for a class of micro-aerial vehicles (MAVs).

The tools necessary to describe and predict these complicated interactions are highly domain specific. The understanding of how the mechanisms of flight vary across configurations and Reynolds number is still an open issue. Due to the complicated nature of the physics, nonlinear analysis is limited to basic predictions, and numerical methods are needed to model the processes. In recent years, the ability to perform large calculations has expanded the reach of many investigations.

A main goal here is the construction of tools suitable to study the fluid-structure interactions (FSI) of highly flexible bodies in low Reynolds number flows. The target application for these tools is the study of insect hovering. These wings are largely considered to be passive structures, and they undergo very large deformations. This generates a very interesting problem for modeling. Extensions of this work could also be toward understanding of fish swimming or any other low Reynolds number flows with passively flexible bodies.

Two-dimensional (2D) models are introduced and analyzed first. The coding

and analysis performed for the 2D cases provide a testing ground for three-dimensional (3D) models, both in terms of the physics and also in the implementation issues. The 2D results provide some tantalizing clues about increased aerodynamic efficiency. A novel aspect of this dissertation is the presentation of a viable 3D FSI code that is practical for a designer to explore flapping dynamics of highly flexible and highly detailed bodies.

1.2 Prior work

Over the past decades, researchers from many fields have studied the flapping flight of insects and birds. The focus of the research varied depending on the interest and background of those involved. Engineers tend to try to mimic flight in an artificial system like a MAV. Scientists tend to be concerned with the fundamental physics involved in the flapping itself, and biologists are focused on explaining how and why particular animals perform what they do. There is immense value in the points of view from each of these disciplines. Each can contribute to the overall understanding of flapping systems and how it relates to natural fliers. The aim of most of these studies has been to understand the complex, unsteady mechanisms that enable the generation of aerodynamic forces for hovering and maneuvering. Insect wings are complex structures that, during flapping, undergo deformations due to aerodynamic, inertial, and elastic forces. To a large extent, the wing's behavior depends on the internal distribution of compliant components and mechanisms (Wootton, 1999). It is important to note that insect wings lack internal muscles and therefore have no actuators to realize internal control forces (Wootton, Herbert, Young, and Evans, 2003). This makes them ideal candidates for engineering synthesis since a vehicle design and control would be much simpler than avian flight.

Experimentally, the landmark papers of Ellington (1984a,b,c,d,e,f) represent the modern era of investigations into flapping insect flight. High speed filming was

used to capture the kinematics of flapping insects. The use of dynamically scaled robots has also been a popular experiment. The investigations of Ellington, van den Berg, Willmott, and Thomas (1996); Dickinson, Lehmann, and Sane (1999) and Sane and Dickinson (2001) all employed a rigid wing to study the effects of basic flapping kinematics. Tethered and free flying animals have also been investigated (Willmott and Ellington, 1997; Spedding, Rosén, and Hedenström, 2003; Fry, Sayaman, and Dickinson, 2005).

The study of flexible bodies as wings is relatively recent. The studies of Combes and Daniel (2003a,b,c) are a highly cited example, wherein direct engineering measurements of the wings of the hawkmoth *Manduca sexta* have been made. Static experiments provided the basis for models for the distribution of material properties of the wing. These experiments were later expanded by Mountcastle and Daniel (2009). As measurement technology improved the deformation that could be measured during the flight of certain insects. The studies of Walker, Thomas, and Taylor (2009c,b,a) used photogrammetry to capture the deflection of very small wings during flight. In Koehler, Wischgoll, Dong, and Gaston (2011), a complete visualization of a flow field around a dragonfly was reconstructed via DNS of data captured by high speed cameras. The kinematics of the body and wing were prescribed, but the visualization of the fluid motion is very descriptive. The biologists Hedrick and Daniel (2006) explored the connection to flight control.

For a variety of species, the roles of inertial, elastic, and aerodynamic forces during flapping flight have been the focus of many investigations; see for example the efforts of Ellington (1984b); Ennos (1989); Lehman and Dickinson (1997); Sun and Tang (2002b); Daniel and Combes (2002); Combes and Daniel (2003c) and Song, Wang, Zeng, and Yin (2001). It is difficult to make direct comparisons between the different studies, not only because the studies usually involve different species but also because different approaches have been used to compute the forces. For

example, Combes and Daniel assessed the relative contributions of aerodynamic, inertial and elastic forces to the wing deformation of the *Manduca sexta* (Combes and Daniel, 2003c). They concluded that the wing motion of this particular insect is mostly determined by the wing's inertial and elastic forces with the aerodynamic loads providing dissipation. During hovering, the typical ratio of wing inertial force to aerodynamic force was found to be about seven. This result was obtained by using scaling arguments and assuming a weight balance to get a fluid-force estimate. In other species, this ratio has been found to be much lower. Ennos, for example, showed that for several species of *Diptera*, the magnitudes of inertial bending moments are about twice the magnitude of the aerodynamic moments during harmonic flapping (Ennos, 1989). The analysis was also based on the weight-balance assumption with harmonic kinematics. However, unlike Combes and Daniel (2003c), Ennos considered the effect of the virtual or added mass of the surrounding fluid. It should be noted that in both of these studies (Ennos, 1989; Combes and Daniel, 2003c), the aerodynamic forces are not correctly estimated since the drag component of the fluid force is neglected.

There have also been many studies that have used highly simplified quasi-steady models to achieve some complex goals. Berman and Wang (2007) employed an almost 3D quasi-steady model to optimize the hovering wing kinematics for minimal energy cost. For controller design, Deng, Schenato, Wu, and Sastry (2006a,b); and Schenato (2003) proposed a nonlinear quasi-steady model, employed averaging methods and linearized the system in order to construct a linear quadratic Gaussian regulator. Another popular approach is the split-cycle control strategy of Doman, Oppenheimer, and Sigthorsson (2010). The usual problems with elaborate control schemes are the size, weight, and energy cost of suitable mechanisms in a MAV.

Artificial systems have been explored by Heathcote and Gursul (2007), and these experiments have been extended to flexible plunging plates by Cleaver, Wang, and

Gursul (2013b). Although these experiments are of limited use to directly compare to insect flapping, they do provide interesting data for numerical validation of FSI implementations. Kang, Aono, Cesnik, and Shyy (2011) used these experiments as the basis for their flapping simulations. The experiments of Zhao, Huang, Deng, and Sane (2010) extend the previous experiments of Dickinson to a range of membrane wings. Since only the resultant forces and moments were measured a direct numerical comparison would be difficult.

Numerical studies inspired by insect flapping were, at least at first, primarily concerned with 2D rigid models (Wang, 2000a,b, 2005). In Wang (2000a,b) a body-fitted grid surrounded an ellipse that underwent prototypical hovering motion. In Miller and Peskin (2005), 2D models are used for a novel investigation of the phenomena called clap-and-fling (Weis-Fogh, 1973). This mechanism is thought to be a transient maneuver associated with take off. One of the first examples of a flexible 2D model is Vanella, Fitzgerald, Preidikman, Balaras, and Balachandran (2009), with the notable demonstration that flexibility may improve the efficiency of flapping.

As processing power has proliferated, there have been many rigid 3D wing models such as Sun and Tang (2002a,b), Ramamurti and Sandberg (2002), Wang and Sun (2005), and Kweon and Choi (2010). Each of these were designed to explore the ideal kinematics that were found in previous studies. In one of the last chapters of the dissertation of Vanella (2010), a 3D direct numerical simulation (DNS) is presented on the longitudinal flight of a rigid system model representing the *Musca domestica*.

Flexible 3D modeling began with the use of primitive fluid models. Daniel and Combes (2002) employed a simple scaling argument to predict that for certain types of FSI, a fully coupled flexible wing model would be required. Dai, Luo, and Doyle (2012) model a flexible rectangular wing in hovering over a wide variety of parameters.

The wing is considered to be thin and have uniform material properties. Erzincanli and Sahin (2013) formulate an arbitrary Lagrangian–Eulerian finite volume method and show an example of a pair of flexible *Drosophila* inspired wings in hovering motion. The grid generation of the wing is tightly linked to the fluid grid generation and warping. This could pose a difficulty in model generation if the wing geometry becomes very complicated.

One of the key engineering applications of flapping flight is the construction of MAVs. Research effort in the field cross many disciplines including mechanism synthesis, flight control, sensors, and optimization. Pai, Chernova, and Palazotto (2009) employed a quasi-steady airfoil theory on a body composed of nonlinear beam and membrane finite elements and compared the predictions to data collected from cameras of an actual MAV. Optimization plays a major role in computational design of MAV, and poses one of the largest challenges to the design process. The use of reduced-order methods to build sensitivity predictions provides the basis for this type of design methodology (Stanford, Beran, Snyder, and Patil, 2013). A recent survey focused on MAV related work has been compiled by Shyy *et al.* (2010).

The use of reduced-order models naturally comes with the trade-off of flexibility and generality. The benefits of computational speed permit black-box optimization for design. However with these methods come the risk of results being nonphysical, and producing unexpectedly poor results in regions away from where they have been tuned.

1.3 Objectives

The principal objectives of this work are to build tools suitable to explore flapping flight and to use these tools to understand the means of flight. Specific objectives include the following:

1. *Two-dimensional studies:* Carry out studies with 2D models and analyze system responses at low Reynolds numbers.

2. *Three-dimensional studies:* Formulate a general 3D model to describe a flexible body that can undergo large displacements and large strains. Develop and implement a FSI algorithm to couple a large-scale fluid solver with a solver for a highly flexible body. Apply tools developed to study flapping flight at low Reynolds numbers.

3. *Experimental studies:* Develop an experimental arrangement to characterize the vibratory response of a live insect wing and determine the wing's modal properties.

1.4 Outline

The rest of this dissertation is organized in the following manner. In Chapter 2, 2D models of flapping flight of a flexible structure are described and studied. Two vastly different fluid models are used, the results obtained are compared, and the benefits of each are explored. In Chapter 3, the modeling of 3D structural wings is discussed. Experiments on a living *Manduca sexta* wing are carried out and a novel material model is formulated for use with finite elements. In Chapter 4, a partitioned FSI method that has been specifically designed for use with the FLASH code, a publically available code available from the University of Chicago,[1] and that makes use of the new finite element model of Chapter 3 is presented and discussed. This scheme can be used to study the motions of flexible structures in viscous flows. Numerical studies conducted with the FEM representation are presented in Chapter 5 along with the simulations conducted for a fully-coupled flexible plate with prescribed harmonic oscillations at one boundary. Finally, concluding remarks, summary, and

[1] `http://flash.uchicago.edu/site/flashcode/`

some thoughts for future work are presented in Chapter 6. Appendices that provide some additional technical details and references are included at the end.

Chapter 2

Two-dimensional studies

As a starting place to understand the phenomena of flapping wings, 2D systems are considered. In 2D the complicated 3D motions are idealized to a representative cross-section of a wing, as depicted in Fig. 2.1. This simplification greatly reduces the complexity of the physics, the possible parameter space, and the computational costs. Working with 2D models provides a basis to understand what types of models should be built for the 3D cases studied later.

2.1 Structural models

2.1.1 Initial study on flexibility

In the first model studied, a single measure for the chord-wise deflection is used. Here, there are two rigid elements connected at point b by a torsion spring as shown in Fig. 2.2. This torsion spring is assumed to be linear with respect to the deflection angle α between the two rigid elements. The location of point b is specified by the coordinates (x, y), and θ is the orientation of link B. The center of mass m_i, $i \in \{\text{A,B}\}$, of each link is located a distance η_i from the connection point b. The length of the profile is l.

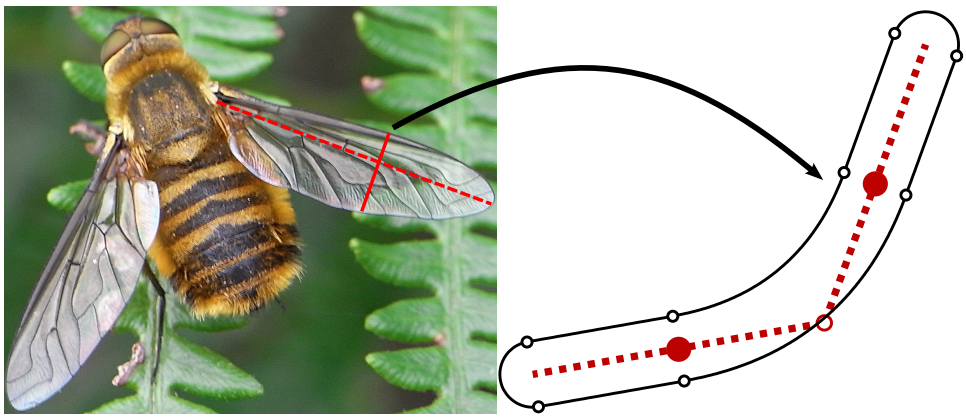
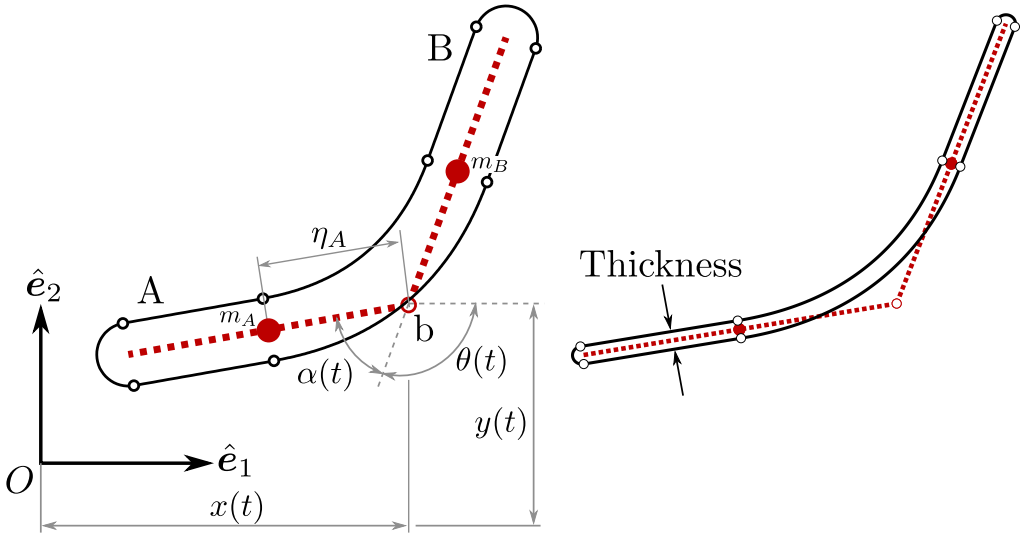


Figure 2.1: Schematic illustration of a two-dimensional wing profile as a representative cross-section of an insect wing. Photo of female *Villa hottentotta*, used under a Creative Commons Attribution-ShareAlike license, accessed from http://commons.wikimedia.org/wiki/File:Villa_hottentotta_female.jpg



(a) Details of the profile's description (b) 3 % Thickness case

Figure 2.2: Schematic of two-dimensional profile with discrete flexibility.

The equations of motion for this structural model can be arranged as

$$\begin{bmatrix} m_A+m_B & 0 & m_B\eta_B\sin\theta - m_A\eta_A\sin(\alpha+\theta) & -m_A\eta_A\sin(\alpha+\theta) \\ & m_A+m_B & m_B\eta_B\cos\theta - m_A\eta_A\cos(\alpha+\theta) & m_A\eta_A\cos(\alpha+\theta) \\ & & I_A+I_B & I_A \\ \text{symm.} & & & I_A \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \\ \ddot{\alpha} \end{bmatrix} = \begin{bmatrix} Q_x + g_x \\ Q_y + g_y \\ Q_\theta + g_\theta \\ Q_\alpha + g_\alpha \end{bmatrix} \quad (2.1)$$

where Q_j, $j \in \{x, y, \theta, \alpha\}$, are the external generalized forces, and g_j are the nonlinear contributions of centrifugal, elastic, and gravity forces. For the case of hovering flight, the motion point b is prescribed. The kinematics of $(x(t), y(t), \theta(t))$ are prescribed functions of time. This reduces the unknowns of (2.1) to a single equation for $\alpha(t)$. This system governs the evolution of the deformation of the profile, and it appears like a nonlinear oscillator

$$I_A\ddot{\alpha} + k\alpha = -I_A\ddot{\theta} + m_A\eta_A\ddot{x}\sin(\alpha+\theta) + Q_\alpha. \quad (2.2)$$

This equation does not contain any structural damping since no assumptions were made as to a particular damping model. A consequence of this choice is that this model cannot be excited at (linear) resonance. Combes and Daniel (2003b,c) proposed certain insects flap their wings near linear resonance, and applied arbitrary proportional damping to make a finite element model response match their data. The damping factor used in the work of Combes and Daniel (2003b) is reported to be 10 times the mass. If that same factor was used to impose linear damping in this model, the system would be over-damped. The damping would dominate the response and the interesting interactions with the fluid would be removed. In addition, structural damping data reported in the literature has been limited, so there is not yet enough

evidence to support a particular material damping model. The current investigation is primarily interested in the elastic response of fully coupled FSI, and significant damping may reduce the influence of the aerodynamic forces making the response structurally dominated. Therefore, structural damping is not considered at this stage but is left as possible future work.

2.1.2 Assumed modes

Since the wing of an insect, or a MAV is a composite structure, an accurate representation calls for a distributed parameter model. However, some discretization is needed to construct a reduced-order model to make computations. A distributed model called the Assumed Modes Method (Meirovitch, 2001) is formulated due to its direct relation to linear vibration theory and its relative simplicity when compared to finite element methods used for moving bodies. The fundamental assumption is that the displacement field v, as depicted in Fig. 2.3, can be approximated as

$$v(\eta, t) = \sum_{r=1}^{n} q_r(t)\varphi_r(\eta). \tag{2.3}$$

This appears like the same expansion as from modal decomposition, so $\{\varphi_r(\eta)\}$ are assumed to be known from vibration theory. Here $\{\varphi_r(\eta)\}$ are taken to be the first r mode shapes of a beam. This approach is similar to the Rayleigh-Ritz method, choosing the weighting functions to be $\{\varphi_r\}$.

The formulation follows a direct construction from Lagrange's equations, where the details of the geometry follow from Fig. 2.3. The position of a point on the deformed beam is given by

$$\boldsymbol{r}(\eta, t) = \boldsymbol{r}_B(t) + \eta\,\hat{\boldsymbol{b}}_1 + v(\eta, t)\,\hat{\boldsymbol{b}}_2. \tag{2.4}$$

The kinetic energy T of the beam is taken to be the following, accounting for

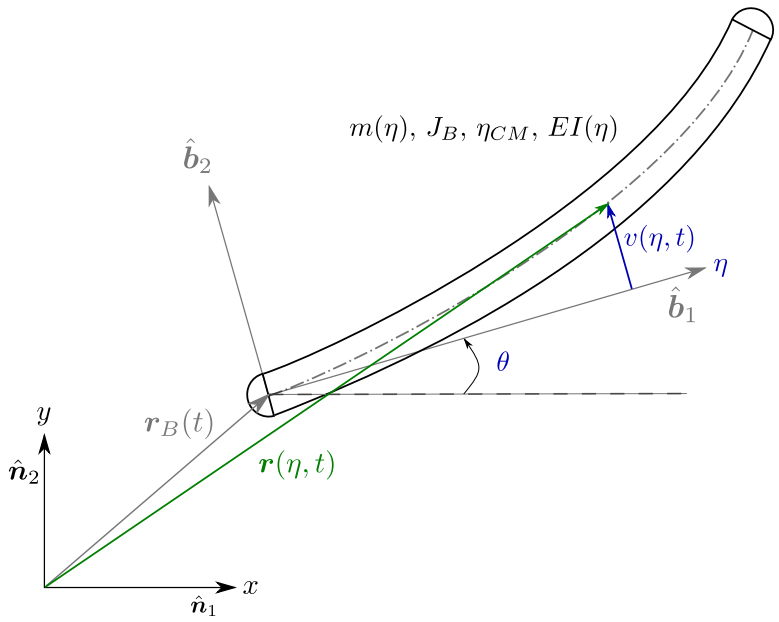


Figure 2.3: Diagram for the formulation of an Euler-Bernoulli beam at an arbitrary location (x, y) and orientation θ in the assumed modes method.

rigid-body motions as well as the deformations.

$$T = \frac{1}{2}\int_0^l m(\eta)\dot{\boldsymbol{r}}(\eta,t)\cdot\dot{\boldsymbol{r}}(\eta,t)\;d\eta \tag{2.5}$$

where $m(\eta)$ is the mass density per unit length. This expression can be expanded

$$\begin{aligned} T = \frac{1}{2}m[\dot{x}^2+\dot{y}^2] + \frac{1}{2}\dot{\theta}^2\int_0^l m(\eta)\,v(\eta,t)^2\;d\eta + \frac{1}{2}J_B\dot{\theta}^2 + \dot{\theta}\int_0^l \eta\,\dot{v}(\eta,t)\,m(\eta)\;d\eta \\ + \frac{1}{2}\int_0^l m(\eta)\,\dot{v}(\eta,t)^2\;d\eta - \dot{\theta}[\dot{x}\cos\theta+\dot{y}\sin\theta]\int_0^l m(\eta)\,v(\eta,t)\;d\eta \\ + [-\dot{x}\sin\theta+\dot{y}\cos\theta]\left[m\,\eta_{CM}\dot{\theta} + \int_0^l m(\eta)\,\dot{v}(\eta,t)\;d\eta\right]. \end{aligned} \tag{2.6}$$

The potential energy, accounting for the elastic potential energy as well as the

gravitational potential energy, assumes the form

$$V = \frac{1}{2}\int_0^l EI(\eta)\left[\frac{\partial^2}{\partial\eta^2}v(\eta,t)\right]^2 d\eta + mgy + mg\eta_{CM}\sin\theta + g\cos\theta\int_0^l m(\eta)\,v(\eta,t)\;d\eta. \quad (2.7)$$

Forming the Lagrangian as $L = T - V$, the Euler-Lagrange equations are formed from the $n+3$ equations

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = Q_x \quad (2.8a)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) - \frac{\partial L}{\partial y} = Q_y \quad (2.8b)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = Q_\theta \quad (2.8c)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_{q_i}, \quad i = 1,\ldots,n. \quad (2.8d)$$

The results, including simplification due to the orthogonality and mass orthonormalization of the mode shapes, are as follows. The resulting coupled system of ordinary differential equations is

$$m\ddot{x} - \ddot{\theta}\left[\cos\theta\sum_{r=1}^{n} b_r q_r + m\eta_{CM}\sin\theta\right] - \sin\theta\sum_{r=1}^{n} b_r\ddot{q}_r + \dot{\theta}^2\left[\sin\theta\sum_{r=1}^{n} b_r q_r - m\eta_{CM}\cos\theta\right] - 2\dot{\theta}\cos\theta\sum_{r=1}^{n} b_r\dot{q}_r = Q_x. \quad (2.9a)$$

$$m\ddot{y} - \ddot{\theta}\left[\sin\theta\sum_{r=1}^{n} b_r q_r - m\eta_{CM}\sin\theta\right] + \cos\theta\sum_{r=1}^{n} b_r\ddot{q}_r - \dot{\theta}^2\left[\cos\theta\sum_{r=1}^{n} b_r q_r + m\eta_{CM}\sin\theta\right] - 2\dot{\theta}\sin\theta\sum_{r=1}^{n} b_r\dot{q}_r = Q_y - mg. \quad (2.9b)$$

$$\ddot{x}\left[-\cos\theta\sum_{r=1}^{n}b_r q_r - m\eta_{CM}\sin\theta\right] + \ddot{y}\left[-\sin\theta\sum_{r=1}^{n}b_r q_r + m\eta_{CM}\cos\theta\right] + \ddot{\theta}\left[J_B + \sum_{r=1}^{n} q_r^2\right] + \sum_{r=1}^{n} a_r\ddot{q}_r + 2\dot{\theta}\sum_{r=1}^{n} q_r\dot{q}_r + mg\eta_{CM}\cos\theta - g\sin\theta\sum_{r=1}^{n} b_r q_r = Q_\theta. \quad (2.9c)$$

and the i-th equation for the deformation q_i is

$$-\ddot{x}b_i\sin\theta + \ddot{y}b_i\cos\theta + a_i\ddot{\theta}_i + \ddot{q}_i - \dot{\theta}^2 q_i + \omega_i^2 q_i + gb_i\cos\theta = Q_{q_i}; \quad i = 1, \ldots, n. \quad (2.9d)$$

The constants are computed from the known φ_r as

$$m = \int_0^l m(\eta)\, d\eta, \qquad a_r = \int_0^l \eta\, m(\eta)\varphi_r(\eta)\, d\eta \quad (2.10)$$

$$\omega_r^2 = \int_0^l EI(\eta)\left[\frac{d^2\varphi_r}{d\eta^2}\right]^2 d\eta, \qquad b_r = \int_0^l m(\eta)\varphi_r(\eta)\, d\eta \quad (2.11)$$

The generalized forces Q_j, $j \in \{x, y, \theta, q_i\}$, are computed from the virtual work performed by the resultant external force over the virtual displacement of the beam. These forces constitute the fluid forces from the fluid-structure interactions. Placing this set of equations in a matrix form in the style of (2.1), gives the mass matrix

$$\mathbf{M}(x, y, \theta, q_1, \ldots, q_n) = \begin{bmatrix} m & & & & & & \text{symm.} \\ 0 & m & & & & & \\ (-\psi_q\cos\theta - m\eta_{CM}\sin\theta) & (m\eta\cos\theta - \psi_q\sin\theta) & J_B & & & & \\ -b_1\sin\theta\, q_1 & b_1\cos\theta\, q_1 & a_1 & 1 & & & \\ \vdots & \vdots & \vdots & 0 & \ddots & & \\ -b_n\sin\theta\, q_n & b_n\cos\theta\, q_n & a_n & 0 & \cdots & 1 \end{bmatrix} \quad (2.12)$$

where $\psi_q = \sum_r b_r q_r$. Since that this matrix is symmetric, it is an indication that the formulation is consistent. The assumed modes are taken to those of a linear

Euler-Bernoulli beam (Meirovitch, 2001) and each mode has the form

$$q_r(\eta) = A_r \sin(\beta_r \eta) + B_r \cos(\beta_r \eta) + C_r \sinh(\beta_r \eta) + D_r \cosh(\beta_r \eta)$$

where the constants β_r, A_r, B_r, C_r, and D_r are found by applying the boundary conditions for a particular beam configuration. The configuration of interest is a prismatic cantilever beam, with standard fixed-free boundary conditions

$$v(0,t) = 0 \qquad \left.\frac{\partial^2 v}{\partial \eta^2}\right|_{\eta=L} = 0$$
$$\left.\frac{\partial v}{\partial \eta}\right|_{\eta=0} = 0 \qquad \left.\frac{\partial^3 v}{\partial \eta^3}\right|_{\eta=L} = 0$$

The constants β_r, A_r, B_r, C_r, D_r, a_r, and b_r are computed in Mathematica by using 50 digits of working precision. This process is required due to the stiffness of the equations that define these factors. The values of ω_r are prescribed in relation to the chosen driving kinematics.

To prototype these equations, the model has been implemented in MATLAB$^{\text{TM}}$ without the fluid coupling. The deformation of a representative case integrated in MATLAB is shown in Fig. 2.4, with a three-mode construction.

2.2 Fluid-structure interactions

Computing the dynamics of the fluid field was performed using two methods in conjunction with the structural model defined in 2.1.1. The first was direct numerical simulation (DNS) of the Navier-Stokes equations for incompressible flow, and the second fluid-model used was the unsteady vortex lattice method (UVLM).

The coupling scheme is a predictor-corrector method (Preidikman, 1998; Yang *et al.*, 2008), as outlined in Fig. 2.5. Although not monolithic, the main benefit to this strategy is that fluid model and structural models are arbitrary. The fluid is

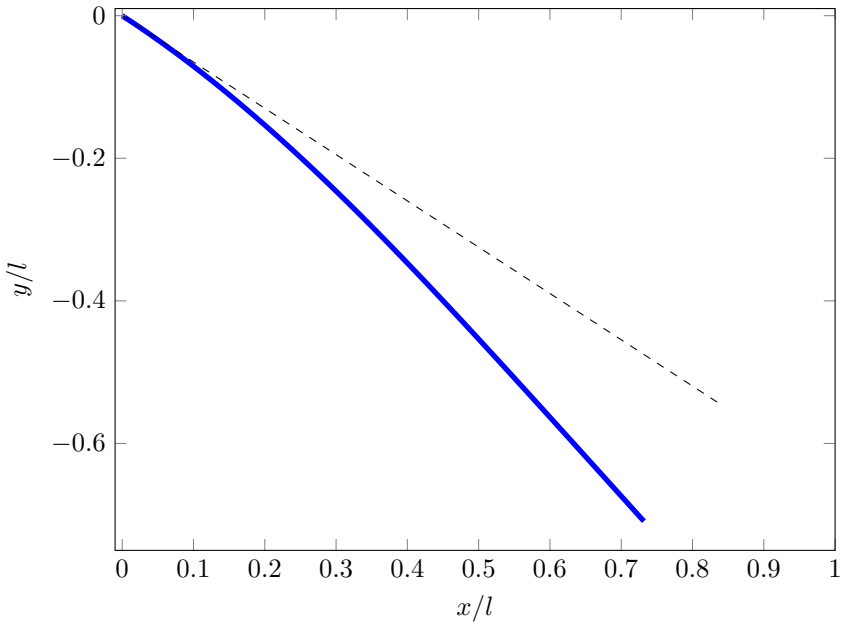


Figure 2.4: Representative results obtained on the basis of the assumed modes method without fluid forces, integrated in MATLAB; --- rigid body frame, —— deformed cantilever beam.

coupled to the structure by computing the resultant forces from the pressure and vorticity on the surface of the body. This force is then used to integrate a predicted set of structural states. In this predicted configuration, the surface kinematics of the body are fed back as the immersed boundary conditions to the fluid model. This cycle of communication is iterated until the change between substeps is below a set tolerance. In all the models used, only 1 or 2 sub iterations are needed. This is likely due to very small time steps used to comply with the Courant-Friedrichs-Lewy condition (CFL) number. This method provides a systematic method to couple the equations of motion in strong form.

2.2.1 Parameterization

The kinematics used are based on simple harmonic hovering(Wang *et al.*, 2004). These have been adapted to include a non-impulsive start (Vanella, Fitzgerald,

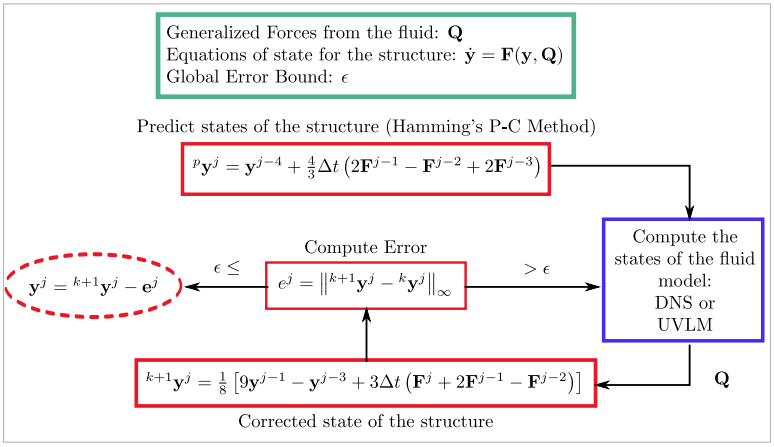


Figure 2.5: Process flow diagram of the fluid-structure coupling scheme.

Preidikman, Balaras, and Balachandran, 2009).

$$
\begin{aligned}
x(t) &= \left(1 - e^{-t/\tau}\right) \frac{A_x}{2} \sin(\omega_f t) \\
y(t) &= 0 \\
\theta(t) &= \theta_0 + \left(1 - e^{-t/\tau}\right) A_\theta \sin(\omega_f t + \phi) \\
\tau &= 0.8 \left(\frac{2\pi}{\omega_f}\right)
\end{aligned}
\tag{2.13}
$$

Here $(x(t), y(t), \theta(t))$ are the location and orientation of the top segment of the profile in Fig. 2.2. The exponential-decay factor is used to prevent the numerical noise from an impulse start. This was found to be beneficial in removing start up noise in the simulations, and having the flow fields remain well behaved through the initial transients. The value of τ is chosen so that it takes around 5 periods of hovering to achieve the regular full motion.

The nondimensionalization of the system's parameters reduces the parametric space to 4 key ratios: $(Re, \frac{\rho_{\text{body}}}{\rho_{\text{fluid}}}, \frac{\omega_f}{\omega_n}, \frac{A_x}{l})$. The reference speed of the Reynolds number $(Re = u_{\text{ref}} l/\nu)$ is chosen to be the peak translation speed of the forcing

$$\max \dot{x} = \frac{A_x \omega_f}{2}.$$

The reference length is the chord l. The ratio between the stroke length and the chord length is taken from Wang *et al.* (2004) to be $\frac{A_x}{l} = 2.8$. The maximum value for rotation is set to $A_\theta = 45°$, and the profile is assumed to rotate about the vertical position $\theta_0 = -\frac{\pi}{2}$. For symmetric hover $\phi = 0$, and there is no lead or lag. Fixing $u_{\text{ref}} = 1$, and $l = 1$ provides a period of $T = 2.8\pi$. The density of the fluid, in the nondimensional DNS code is already 1, so to set the Reynolds number the kinematic viscosity ν is selected. The parameters of the structure (η_A, m_A, I_A, k) from (2.1) are computed by geometry and choice of ($\frac{\rho_{\text{body}}}{\rho_{\text{fluid}}}$, $\frac{\omega_f}{\omega_n}$). From Fig. 2.2, each rigid link is taken to be a rectangle with a circular endcap. Due to computational considerations in the fluid solver, a finite thickness profile is needed. The area, location of the centroid η_A, mass m_A, and rotary moment of inertia I_A can be directly computed from geometry once the density ratio is chosen. Finally, the spring constant k is computed from the relation

$$\omega_n^2 = \frac{k}{I_A}\,. \tag{2.14}$$

In the next subsections, the results obtained through the DNS and UVLM computations are introduced and discussed. There the ratio $\rho_{\text{body}}/\rho_{\text{fluid}} = 25$ has been chosen to scale the aerodynamic forces to be of the same order as the fluid forces. The fluid-structure interactions are investigated by selecting various spring constants in the model. As discussed above, the difference spring values correspond to different choices of the ratio ω_f/ω_n. In the following results, the values of ω_f/ω_n range from the soft case corresponding to the ratio of 1/2, to the intermediate spring constants of 1/3 and 1/4, and the almost rigid spring case corresponding to $\omega_f/\omega_n = 1/6$.

2.2.2 Direct numerical simulations in two dimensions

Direction numerical simulation (DNS) represents the highest fidelity computational fluid model in common use. It is constructed by the direct discretization on a staggered grid of the Navier-Stokes equations for incompressible flow. The results, as presented in Vanella *et al.* (2009), are constructed from a second order central difference scheme on a stretched Cartesian mesh. Time marching is performed by using the fractional step method (Kim and Moin, 1985). The body is represented in the fixed grid by using an immersed boundary method (Yang *et al.*, 2008). To enforce the no-slip condition, the predicted velocity field of the fractional step method is forced to match the velocity field along the surface of the body. The flapping profile is placed in the center of a large box so that the boundaries do not interact with the body. The equations are integrated from rest to 15 periods of motion. It takes approximately 1.5 days to compute a single period T of flapping motion on a Intel XEON based computer.

The computational grid was constructed to resolve the boundary layers, and other flow features on the moving profile. A schematic of the domain is shown in Fig. 2.6 with an an expanded view near the tip of the body. The center point of the profile is located at the center of a $30l \times 30l$ domain to minimize the effects of the far-field boundary conditions. The center region, where the body passes through, is a uniform grid with cell size $\Delta x = \Delta y = 3.725 \times 10^{-3} l$. This provides approximately 8 or 16 points inside the boundary for the various Reynolds number cases. Outside of this region, the grid is stretched to the boundaries.

A summary of the resulting flow fields is depicted in Fig. 2.7 for a range of ω_f/ω_n at Reynolds number $Re = 75$. The vorticity contours reveal the vortex structure interplay with the flexible profile. Similar results were also computed for $Re = 250$ and $Re = 1000$. It is worth noting that for the soft spring case of $\omega_f/\omega_n = 1/2$, the system undergoes extremely large deflections, and the passive link

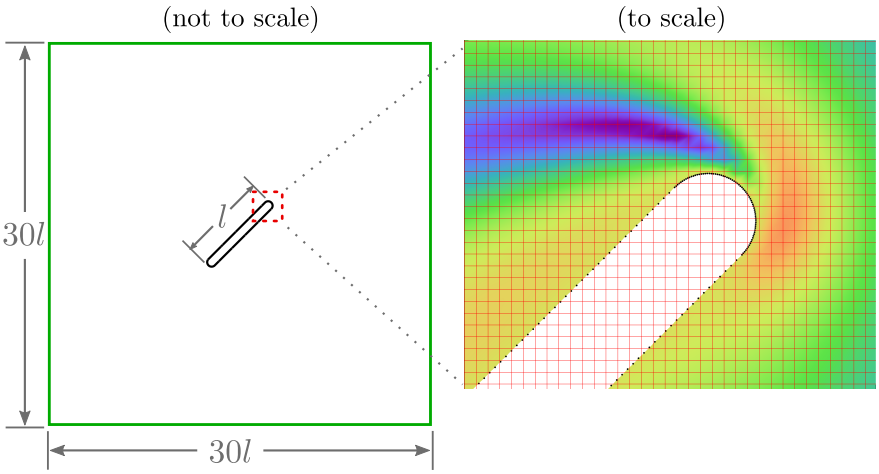


Figure 2.6: Schematic of the profile inside the DNS grid showing the overall size of the domain. The detailed view shows a close up near the leading edge of a 10 % thick body, where the red lines are grid lines, the black points are control points representing the body, and the flow field is the magnitude of velocity for $Re = 250$, $\omega_f/\omega_n = 1/2$, at time $t/T = 9.75$.

almost undergoes a complete rotation about the joint.

Looking at the averaged dimensionless aerodynamic forces acting on the profile, there is a very interesting finding. It is observed that for the particular spring value corresponding to $\omega_f/\omega_n = 1/3$, there is a peak in the ratio of lift coefficient and drag coefficient C_L/C_D, as viewed in Fig. 2.8. It is also noted that the flexible profile has an improved efficiency compared to that of the rigid profile.

2.2.3 Unsteady vortex lattice method in two dimensions

In contrast to the computationally expensive DNS method, vortex methods present a compromise between speed and fidelity. In the unsteady vortex lattice method (UVLM) employed by Preidikman (1998), it is assumed that the flow field is inviscid and the wake can be completely described by point vorticies. A body in the flow field is discretized into panels, and the no-penetration condition is applied at the chosen control points along each panel. The discretization concepts used in this

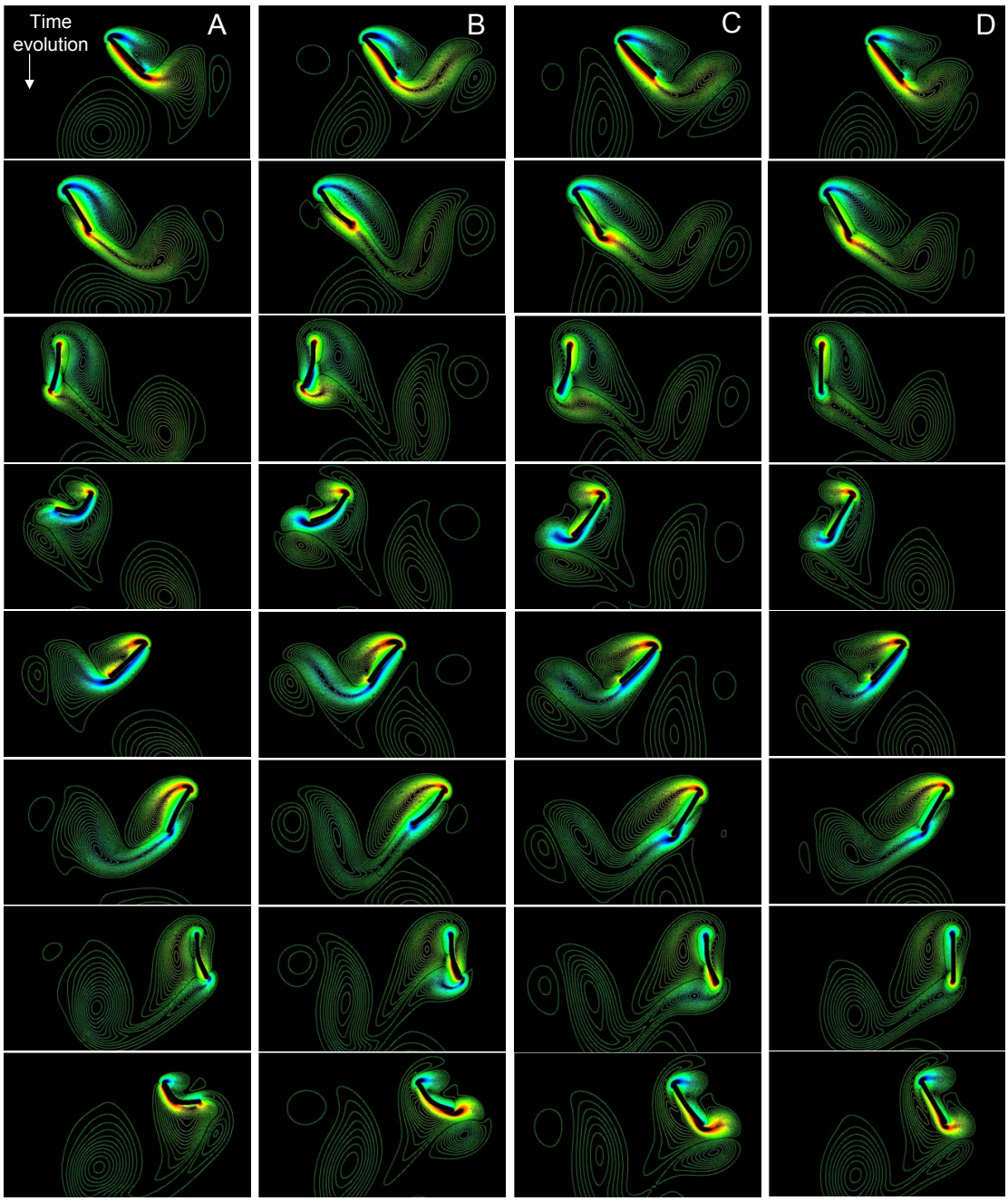


Figure 2.7: Vorticity contours from DNS at $Re = 75$. Contours range from -10 to 10 with 80 intervals. Columns A-C are of flexible profiles with $\omega_f/\omega_n = 1/2,\ 1/3,\ 1/4$ respectively; Column D is of the rigid profile. Adapted from Vanella *et al.* (2009).

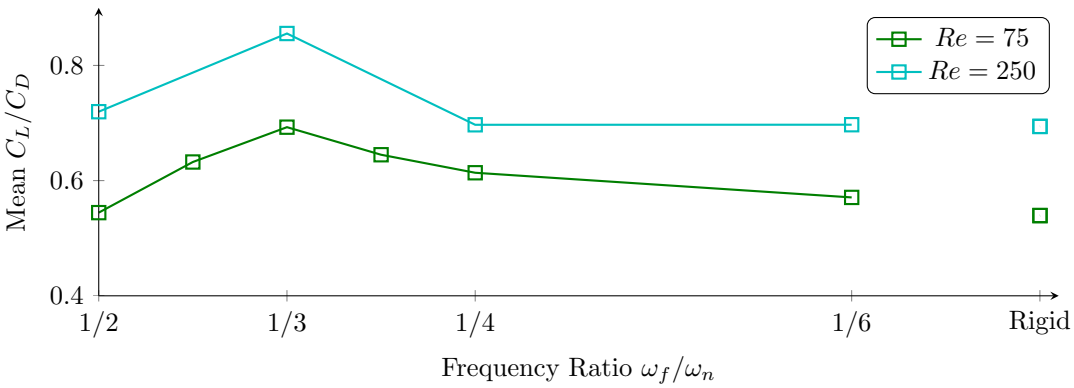


Figure 2.8: Averaged lift to drag ratio from the two-dimensional DNS calculations for various frequency ratios. Profile thickness is $l/10$.

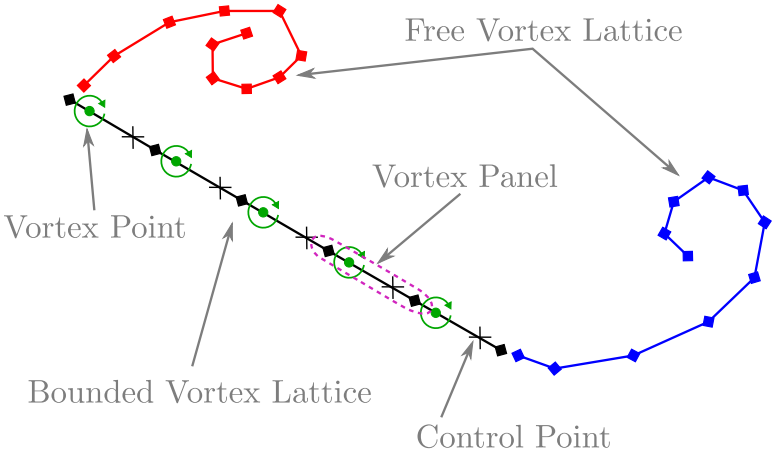


Figure 2.9: Schematic for the discretization of the Unsteady Vortex Lattice Method.

method are illustrated in Fig. 2.9.

This method can be applied to a membrane or zero-thickness problems, since assuming the wake to only separate at the edges is suitable. Vortices are convected from the trailing edge after each time step by using the Kutta condition. Similarly, Valdez, Preidikman, and Massa (2006) proposed a method to convect vortices simultaneously from the leading of the profile. Since a zero-thickness profile is used here, then the tips are assumed to be the points where the wake separates from the body. Also proposed by Valdez *et al.* (2006), is a reconstruction of the entire velocity and pressure fields from the vortex particle wake. For the results presented here, the

original code was authored by Valdez (2008).

Unlike the slower DNS computations, a complete run of 20 periods of hovering motion takes around 10 hours on an eight processor Intel XEON computer. The issue with longer simulations is that each additional time step, there are two additional vortices whose influence needs to be included. So the number of computations increases at a rate $O(n^2)$, where n is the number of time steps in the integration. This makes short calculations very quick in comparison to DNS, but long time simulations quickly become impractical. These characteristics make the UVLM attractive from a design perspective since coarse results can be obtained rather quickly.

In this study, the assumed mode method based beam equations presented in Section 2.1.2 have been coded into a UVLM solver. As shown in Figure 2.10, with the vortex particles being convected from the leading and trailing edges, the profile deforms. These equations proved to be troublesome due to their stiffness in integrating, providing unrealistic dynamics if the deformation became large, and inherent limitation of the flexibility of the profile. However this does provide some insight into the need to use a large-deformation nonlinear formulation to achieve something more realistic.

2.3 Identification and characterization

The computational models described above are used to explore key phenomena associated with the motions of a flexible hovering system. Three main tools are employed. First, the structural resonance of the body is investigated numerically using continuation. Second, the dimension of the coupled FSI system is estimated. Third, the proper orthogonal decomposition is used to construct empirical modes of the fluid domain.

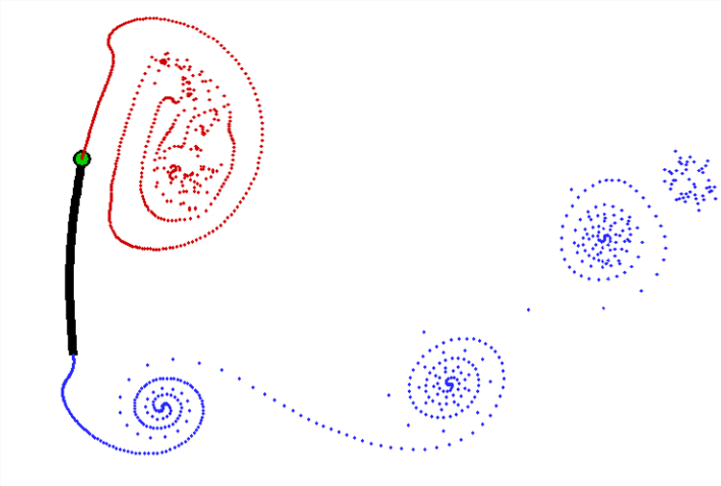


Figure 2.10: Assumed mode method model in UVLM fluid model, vortex particles are convected from the leading edge (red), and the trailing edge (blue).

2.3.1 Structural resonance

The efficiency increase that was observed for certain moderate stiffness values invites further investigation as to the behavior of the profile. It will be shown that in the absence of the fluid the profile displays a nonlinear resonance near $\omega_f/\omega_n = 1/3$. This amplification of the camber at this superharmonic falls in the predicted range that most natural fliers flap around 1/2 the first natural frequency (Kang, Aono, Cesnik, and Shyy, 2011). The study is performed numerically since the analytic forms do not easily admit asymptotic approximations for the given parameter values.

2.3.1.1 Transformation to an autonomous system

The numerical continuation of limit cycles usually involves autonomous systems. This presents a problem, since the forcing terms from the kinematics explicitly depend on time. The simple embedding of time t as an additional state will not produce a limit cycle in all the states and this poses problems for numerical methods. Instead, the normal-form of a Hopf bifurcation(Nayfeh and Balachandran, 1995) can be used

since the forcing terms are harmonic. This method provides an asymptotically stable way to harmonically force the profile. The transformation is started by taking (2.2), and replacing $\theta(t)$ and $x(t)$ with the harmonic forcing functions. For numerical stability, a linear damping term is added.

$$\begin{aligned} I_A\ddot{\alpha}(t) + c\dot{\alpha}(t) + k\alpha(t) = &-I_A\left(-\omega_f^2 A_\theta \sin(\omega_f t)\right) \\ &+ m_A\eta_A\left(-\omega_f^2\frac{A_x}{2}\cos(\omega_f t)\right)\sin(\alpha(t) + \theta_0 + A_\theta\sin(\omega_f t)) \end{aligned}$$

This equation can be simplified since $\theta_0 = -\pi/2$, and the following parameters can be defined.

$$\omega_n = \sqrt{k/m} \tag{2.15a}$$

$$\zeta = \frac{c}{2I_A\omega_n} \tag{2.15b}$$

$$b = \frac{m_A\eta_A}{I_A}\frac{A_x}{2} \tag{2.15c}$$

$$\begin{aligned} \ddot{\alpha}(t) + 2\omega_n\zeta\dot{\alpha}(t) + \omega_n^2\alpha(t) = &\omega_f^2 A_\theta\sin(\omega_f t) \\ &+ b\omega_f^2\cos(\omega_f t)\cos\left(\alpha(t) + A_\theta\sin(\omega_f t)\right) \end{aligned} \tag{2.16}$$

The forcing oscillator that has an asymptotically stable limit cycle takes the form

$$\begin{aligned} \dot{u} &= u + \omega_f v - u\left(u^2 + v^2\right) \\ \dot{v} &= -\omega_f u + v - v\left(u^2 + v^2\right). \end{aligned}$$

This normal-form oscillates with frequency ω_f, and the limit cycle is the unit circle in the phase plane. Replacing the sine and cosine terms in (2.16) with $u(t)$ and

Table 2.1: Parameter values used in continuation of periodic motions

Quantity	**Value**	
A_x/l	2.8	
A_θ	$\pi/4$	rad
ω_f	$2/2.8$	rad/s
ζ	0.01	
b	4.222 96	

$v(t)$, respectively completes the transformation to an autonomous system. In first order form, where $q_1(t) = \alpha(t)$ and $q_2(t) = \dot{\alpha}(t)$, gives the equation structure that is suitable for use in numerical methods.

$$\dot{q}_1 = q_2 \tag{2.18a}$$

$$\dot{q}_2 = -\omega_n^2 q_1 - 2\omega_n \zeta q_2 + \omega_f^2 \left(A_\theta u + bv \cos\left(q_1 + A_\theta u\right)\right) \tag{2.18b}$$

$$\dot{u} = u + \omega_f v - u\left(u^2 + v^2\right) \tag{2.18c}$$

$$\dot{v} = -\omega_f u + v - v\left(u^2 + v^2\right) \tag{2.18d}$$

2.3.1.2 Numerical continuation

The finding and continuation of the limit cycles of (2.18) is performed in MATLAB using MatCont (Dhooge, Govaerts, and Kuznetsov, 2003; Dhooge, Govaerts, Kuznetsov, Meijer, and Sautois, 2008). The fixed parameters were chosen to match those used throughout the simulations and are shown in Table 2.1.

The first limit cycle was located by choosing a very stiff spring, $\omega_f/\omega_n = 1/10$, and integrating the solution for several hundred cycles. Once the orbit appears periodic with period $T = 2\pi/\omega_f$, then any point on this cycle can be used as a starting point for the continuation. A pseudo-arclength method (Kuznetsov, 2004) is used to find the limit cycle as ω_n is varied. The results over a broad frequency range are shown in Fig. 2.11 for the amplitude of the steady state oscillation as a

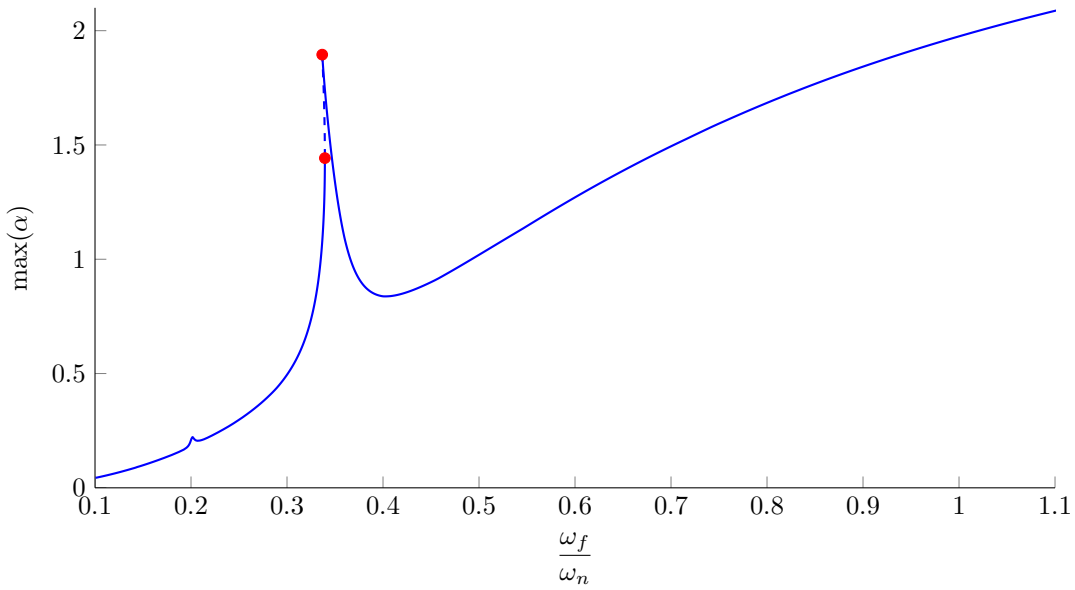


Figure 2.11: Numerical continuation results for two-dimensional profile motions over a broad range of harmonic excitation frequencies. Solid and dashed lines are used to represent stable and unstable periodic motions, respectively.

function of ω_f/ω_n.

Zooming into the region around $\omega_f/\omega_n = 1/3$ shows that the peak appears like a softening hook. An expanded view around the nonlinear resonance region is provided in Fig. 2.12. The red dots are used to identify the turning points in this response graph. When exploring with MatCont, it was found that the b forcing term shifts the principal peak to 1/3, but it requires the additional linear forcing to make the resonance have the hook shape. The use of larger values of ζ quickly destroy the nonlinear resonance.

It is unusual that this configuration does not have the expected resonance at $\omega_f/\omega_n = 1$. Future efforts could include studying the influences of alternate kinematics on the resonance locations (including those for nonlinear resonances) as well as the relation to the aerodynamic efficiency. The functions used in Berman and Wang (2007) would serve as a general framework for a variety of flapping styles and are continuously differentiable. To compare them to the piecewise kinematics

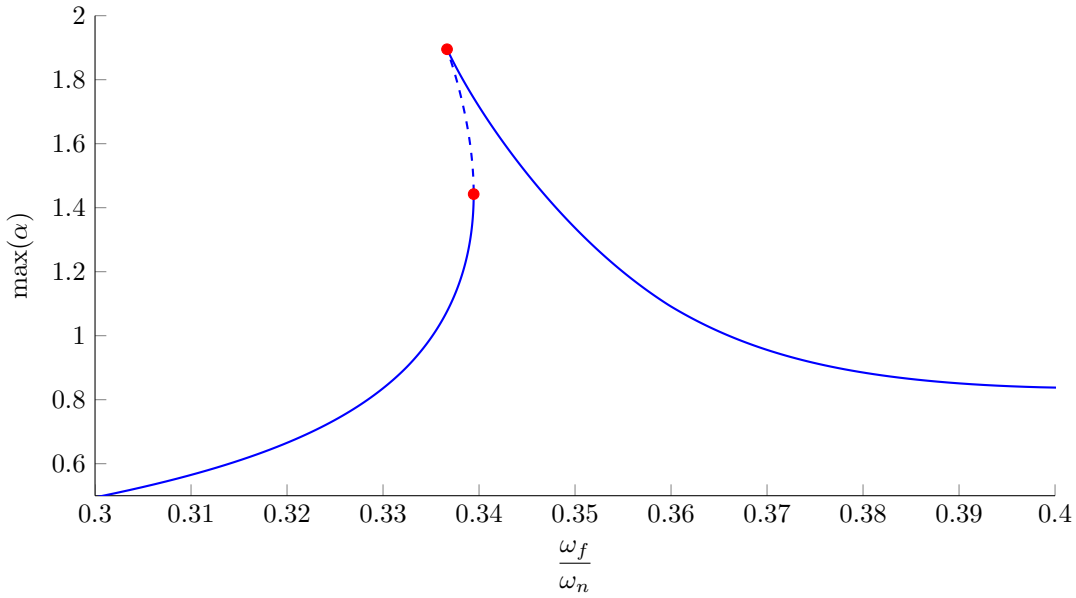


Figure 2.12: An expanded view around the nonlinear resonance provided in Fig. 2.11.

implemented in the 2D code, which are representations of the curves presented by Dickinson *et al.* (1999), some modifications may be required.

2.3.2 Dimension calculations

As an attempt to uncover underlying invariants that both the DNS and UVLM simulations might share when the same structural model is used, estimates of the dimensions of the responses has also been carried out. This idea comes out of the geometric notions of fractals (sets with non-integer dimensions), and strange attractors (attracting sets with non-integer dimensions). Farmer, Ott, and Yorke (1983) gives the interpretation that dimension relates the amount of information needed to specify the position of a point on the attractor to within a given level of accuracy. The construction of the method follows Nayfeh and Balachandran (1995). When the correlation dimension D_2 is approximately one, the corresponding orbit is periodic, and when D_2 is approximately two this indicates a quasiperiodic orbit. If D_2 is not an integer but a real number, the corresponding object can be assumed to

be a fractal. When D_2 is in between one and two, this may imply chaotic motions.

2.3.2.1 Formulation

Let $s(t)$ be the generic signal of interest that maps $s : \mathbb{R} \to \mathbb{R}$, so it could be α, C_L, or C_D from the flexible profile model. First, this signal is unfolded in pseudo-state space in order to define the size of the space in which the attractor lies on. This is performed by constructing the so-called delay coordinates with delay τ. The position in the new state space of dimension d is defined as

$$\mathbf{x} = \begin{bmatrix} s(t) & s(t+\tau) & s(t+2\tau) & \cdots & s(t+(d-1)\tau) \end{bmatrix}^T. \tag{2.19}$$

The time delay τ is found by constructing the autocorrelation function of the sampled data set, $s_i = s(t_0 + i\Delta t)$ where Δt is the sampling period and $i = 1, 2, \ldots, N$. The first zero crossing, if it exists, from the autocorrelation function is used as the delay τ. If there is no crossing, then the first minimum is used. This provides a linear independence between the signals $s(t)$ and $s(t+\tau)$.

There are many measures of the dimension of an attractor, such as the Capacity, Pointwise, Information, Correlation, and Lyapunov Dimensions. Each measure has its strengths, such as estimation of the dimension from other quantities such as Lyapunov exponents, or data set reduction by random sampling. Here the Correlation Dimension is chosen due to its computational simplicity, and the entire signal data set is used.

From data set $\mathbf{x}_i = \mathbf{x}(t_i) \in \mathbb{R}^d$, $i = 1, 2, \ldots, N$, an estimation of the correlation dimension is based on to how many other points of the signal are within a sphere of radius r. The correlation can be computed as

$$C(r) := \lim_{N\to\infty} \left[\frac{1}{N^2} \sum_{i,j=1}^{N} H\left(r - \|\mathbf{x}_i - \mathbf{x}_j\|_2\right) \right] \tag{2.20}$$

where H is the Heaviside step function

$$H(r) = \begin{cases} 0, & \text{if } r < 0 \\ 1, & \text{if } r \geq 0. \end{cases} \tag{2.21}$$

The correlation dimension, denoted D_2, is defined as

$$D_2 := \lim_{r \to 0} \frac{\ln C(r)}{\ln r}. \tag{2.22}$$

For numerical robustness, r is chosen to be finite, and $\ln C(r)$ is computed for a range of r and d. A plot of $\ln C(r)$ versus $\ln r$ is built and used to estimate the slope. An intermediate range of r where the slope approaches a constant as d increases is identified, and the slope in this region is used to estimate D_2. This procedure gives us two important estimations: d and D_2. Knowing d informs us as to the needed number of delay coordinates that the attractor lives in, while D_2 gives insight into the dimension of the attractor.

2.3.2.2 Results

Choosing $\alpha(t)$ as the signal of interest, the first decision is to choose the time delay τ. Computing the autocorrelation for each parameter configuration, the first zero crossing can be located. In Fig. 2.13, the normalized autocorrelations, computed for the data from the DNS as well as the UVLM simulations are shown. They show close agreement for the values of delay τ, when normalized to the hovering period.

Tabulating the slopes on an intermediate finite range in $\ln r$ is presented in Table 2.2. According to Takens' Embedding Theorem (Nayfeh and Balachandran, 1995), a structure of dimension d_A can be embedded in a space $> 2d_A$. This means that for the stiffer spring value of $\omega_f/\omega_n = 1/4$, the dimension d was not varied enough to capture D_2. However, looking at the soft spring cases, there is very nice

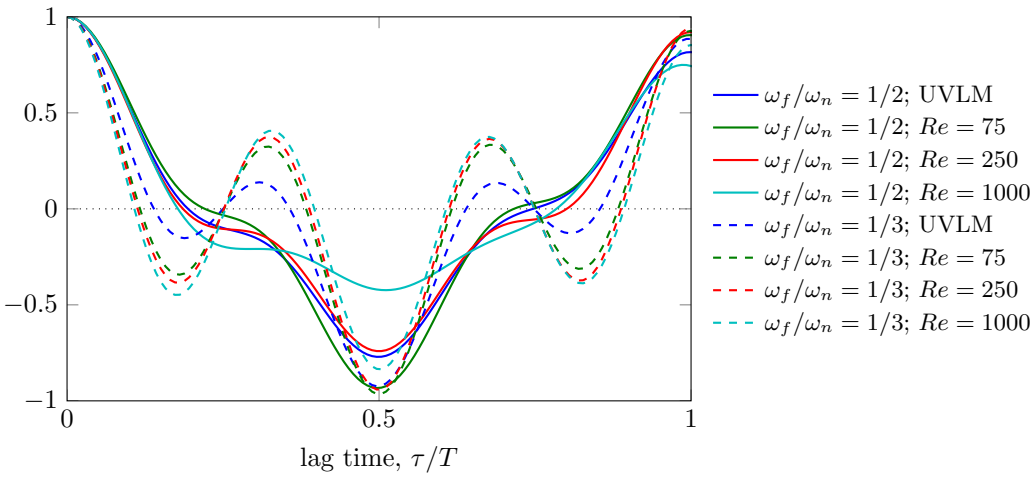


Figure 2.13: Representative normalized autocorrelation of $\alpha(t)$ for various Reynolds numbers and ratios of ω_f/ω_n.

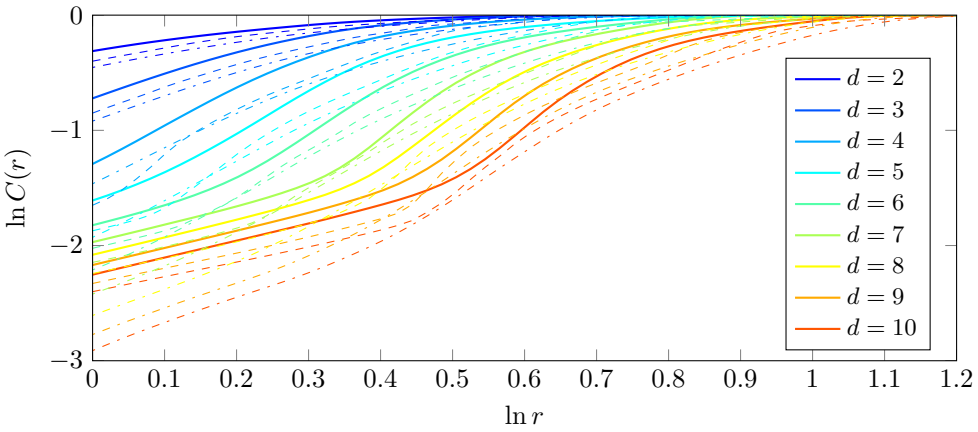


Figure 2.14: Representative correlation dimension calculations of $\alpha(t)$ for $\omega_f/\omega_n = 1/3$. Solid lines (——) UVLM, dashed lines (- - -) DNS at $Re = 75$, dot-dashed lines (·-·-) DNS at $Re = 250$.

Table 2.2: Estimation of the Correlation Dimension D_2 from $\ln r$–$\ln C(r)$ plots.

	$Re = 75$		$Re = 250$		$Re = 1000$		UVLM	
ω_f/ω_n	d	D_2	d	D_2	d	D_2	d	D_2
1/2	7	1.2	5	1.3	6	2.0	4	1.6
1/3	8	1.3	8	1.3	8	2.3	7	1.5
1/4	7	1.2	> 10	> 4	> 10	> 3	8	2.0

agreement among the results from the DNS runs and reasonable agreement with the UVLM results.

2.3.3 POD analysis of the flow fields

2.3.3.1 Formulation

The proper orthogonal decomposition (POD) goes by many names such as the Karhunen-Loéve transform, principal component analysis, or singular systems analysis depending on the discipline. It also can be formulated in a continuous or discrete sense, and used for experimental or computational data. The overarching goal of the POD is to decompose data into hierarchical sets of spatial basis functions, often called mode shapes. Here the velocity fields of the fluid will be decomposed into spatial modes using the continuous approach. The gives the assumed representation of the flow field the following form

$$\boldsymbol{u}(\boldsymbol{x}, t) = \sum_{i=1}^{\infty} \alpha_i(t)\boldsymbol{\phi}_i(\boldsymbol{x}). \tag{2.23}$$

This can be interpreted from a vibrations perspective as time-dependent coefficients in modal coordinates. The modes $\boldsymbol{\phi}(\boldsymbol{x})$ are chosen to maximize the projection of the empirical data onto these modes in a L^2 sense. So the problem statement is

formulated as in Holmes, Lumley, and Berkooz (1996), in the scalar case as

$$\max_{\phi \in L^2[0,1]} \frac{\left\langle |(u,\phi)|^2 \right\rangle}{\|\phi\|^2}$$

where $(\cdot,\cdot)$ is the L^2 inner product on the domain of interest, and $\langle\cdot\rangle$ is some averaging operation on the ensemble of data. This all implies that ϕ is chosen to maximize the energy contained in the original data. Casting the solution as a variatonal problem and imposing the constraint that each mode is normalized, provides the so-called Fredholm equation

$$\int_\Omega R(x,s)\phi(s)\ ds = \lambda\phi(x).$$

Here $R(x,s) = \langle u(x,t) \otimes u(s,t) \rangle_t$ is the time averaged autocorrelation tensor of the field, ϕ are the unknown mode shapes, and λ are associated eigenvalues. Further simplification can be made, since it can be recognized that the modes are a special superposition of the data snapshots. Employing the *method of snapshots* (Sirovich, 1987), the modes ϕ can be approximated as a finite sum over the known data

$$\boldsymbol{\phi}_i(\boldsymbol{x}) = \sum_{k=1}^{M} \psi_{ik} \begin{bmatrix} u(\boldsymbol{x},t_k) \\ v(\boldsymbol{x},t_k) \\ p(\boldsymbol{x},t_k) \end{bmatrix}. \tag{2.24}$$

O'Donnell and Helenbrook (2007) demonstrate through a scaling argument that the pressure components can be neglected for incompressible flow in the substitution back into the Fredholm equation. This means that only the velocity field is needed for the computations, but the modes of the pressure field are still computed. The simplified results become an algebraic eigenvalue problem of size M

$$\mathbf{C}\boldsymbol{\psi} = \lambda\boldsymbol{\psi} \tag{2.25}$$

where

$$C_{lk} := \frac{1}{M} \int_{\Omega} \boldsymbol{u}(\boldsymbol{x}, t_l) \cdot \boldsymbol{u}(\boldsymbol{x}, t_k)\, dV. \tag{2.26}$$

Since $\mathbf{C} = \mathbf{C}^T$ only the upper or lower triangular part needs to be constructed. So the process to construct the POD set can be described as a sequence of the following steps:

1. Generate velocity field data at equal time intervals.
2. Construct each element of (2.26) by integrating over the domain.
3. Solve the $\mathbb{R}^{M \times M}$ algebraic eigenvalue problem of (2.25) to get the set of eigenvalues $\boldsymbol{\lambda}$ and the associated eigenvectors $\boldsymbol{\psi}$.
4. Back substitute $\boldsymbol{\psi}$ into (2.24) to get a truncated set of POD eigenfunctions $\boldsymbol{\phi}$.

The set of spatial mode shapes $\boldsymbol{\phi}$ are ranked by using the eigenvalues $\boldsymbol{\lambda}$. This gives a quantitative measure to regard the importance of each mode as constructed from the data. For all the examples below 99 % of the energy in the snap shots in contained in less then 10 of the leading eigenvalues. This significant roll off means that one can consider just a few leading terms. A noteworthy side effect of the POD, that by Sirovich's method is immediately appreciable, is that ϕ must be divergence free. Since the mode shapes are a weighted super position of divergence free data, then ϕ must also be divergence free (incompressible). So any use of these modes faithfully preserves the incompressibilty of the flow field.

2.3.3.2 Results

Next, the results obtained from various POD computations are presented. Note that these figures represent the flow field outside of time, and they are the hierarchical structures of the fluid flow. The center joint of the profile moves through the region $(x/l,\, y/l) = (\pm 1.4,\, 0)$.

In Fig. 2.15, the vorticity contours of the computed modes are shown for $Re = 75$. The plotted field is the curl of the velocity mode shape, normalized to have the maximal value be one. This makes comparison between modes informative. Looking at this figure, it is observed that all of the mode one results contain a large pair of vortices. This represents the downward jet of fluid, and produces lift on the profile. For $\omega_f/\omega_n = 1/3$, this pair of vortices is the most intense as well as closest to the region of the body. Interestingly this corresponds to the most efficient response for the given harmonic input kinematics. It also corresponds to the structural resonance for the kinematics being used. As expected, the scale of the structures decreases as the mode number increases. This follows intuition from vibration mode shapes.

To contrast the well organized structures seen at $Re = 75$, the results of $Re = 250$ show a slightly different story, see Fig. 2.16. It is interesting to note similar patterns to the lower Reynolds number configuration; again mode ones shows a downward jet and mode two shows the end of stroke vortex pair. In this flow regime, the vortices for the spring of intermediate stiffness ($\omega_f/\omega_n \in \{1/3, 1/4\}$) appears to be more spatially regular. An investigation using other kinematics would need to be done in order to see how that affects the regularity of the field.

Comparing the mode shapes of DNS fields to the UVLM fields cannot be done with vorticity contours since the UVLM is inviscid. Instead the velocity contours must be compared. As discussed later in §2.4.1, the UVLM velocity modes share similar features to the DNS generated modes.

2.4 Comparing the DNS and UVLM models

The discussion in this section follows from the published work of Fitzgerald, Valdez, Vanella, Balaras, and Balachandran (2011). The goal is to compare the DNS and UVLM models to see where and how each should be best employed. A DNS

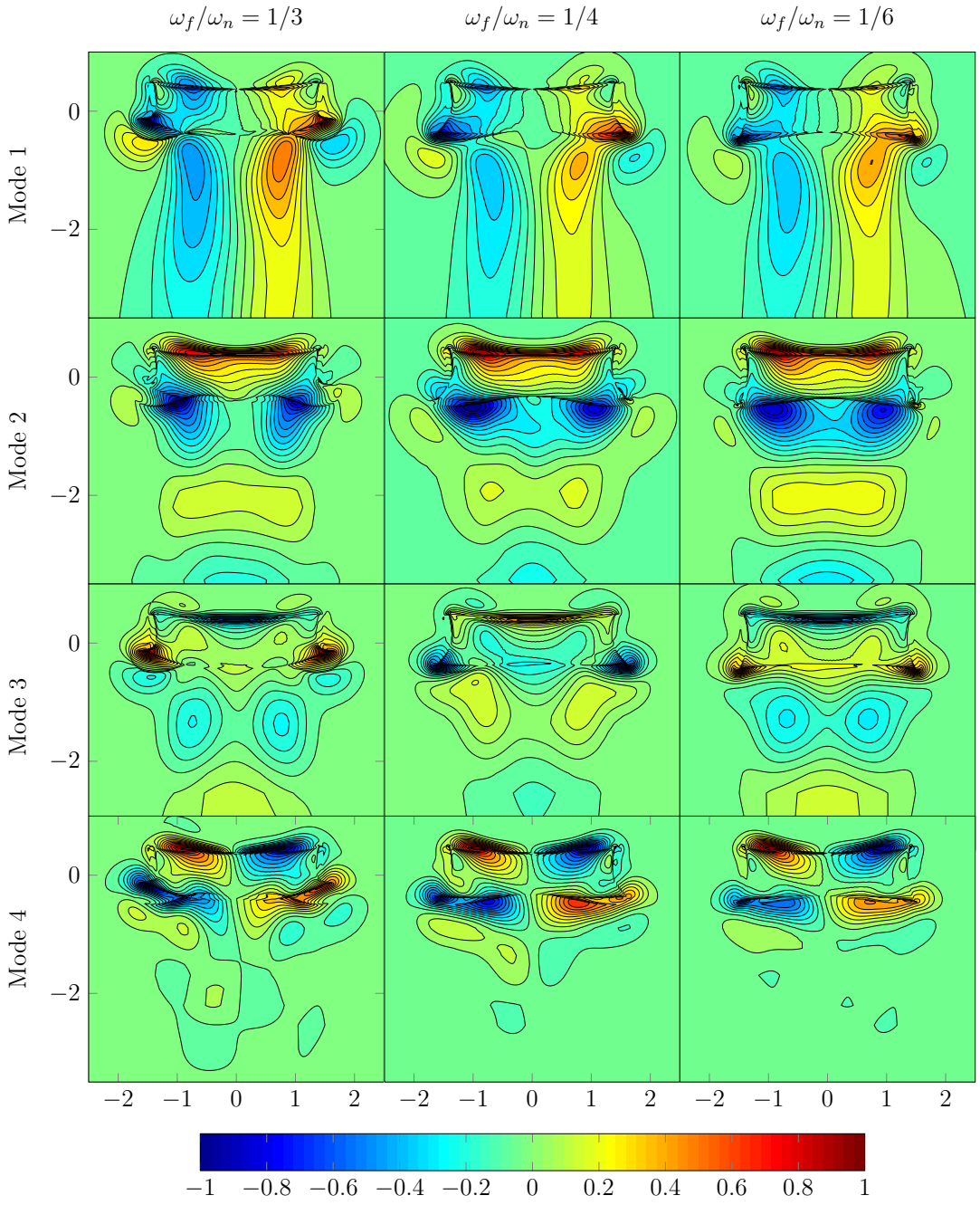


Figure 2.15: Vorticity contours determined by POD from DNS at $Re = 75$, 10 % thickness, and harmonic kinematics. Normalization is $\max_{\mathbf{x}} |\nabla \times \boldsymbol{\phi}_i(\boldsymbol{x})| = 1$. Horizontal scale is x/l, vertical scale is y/l.

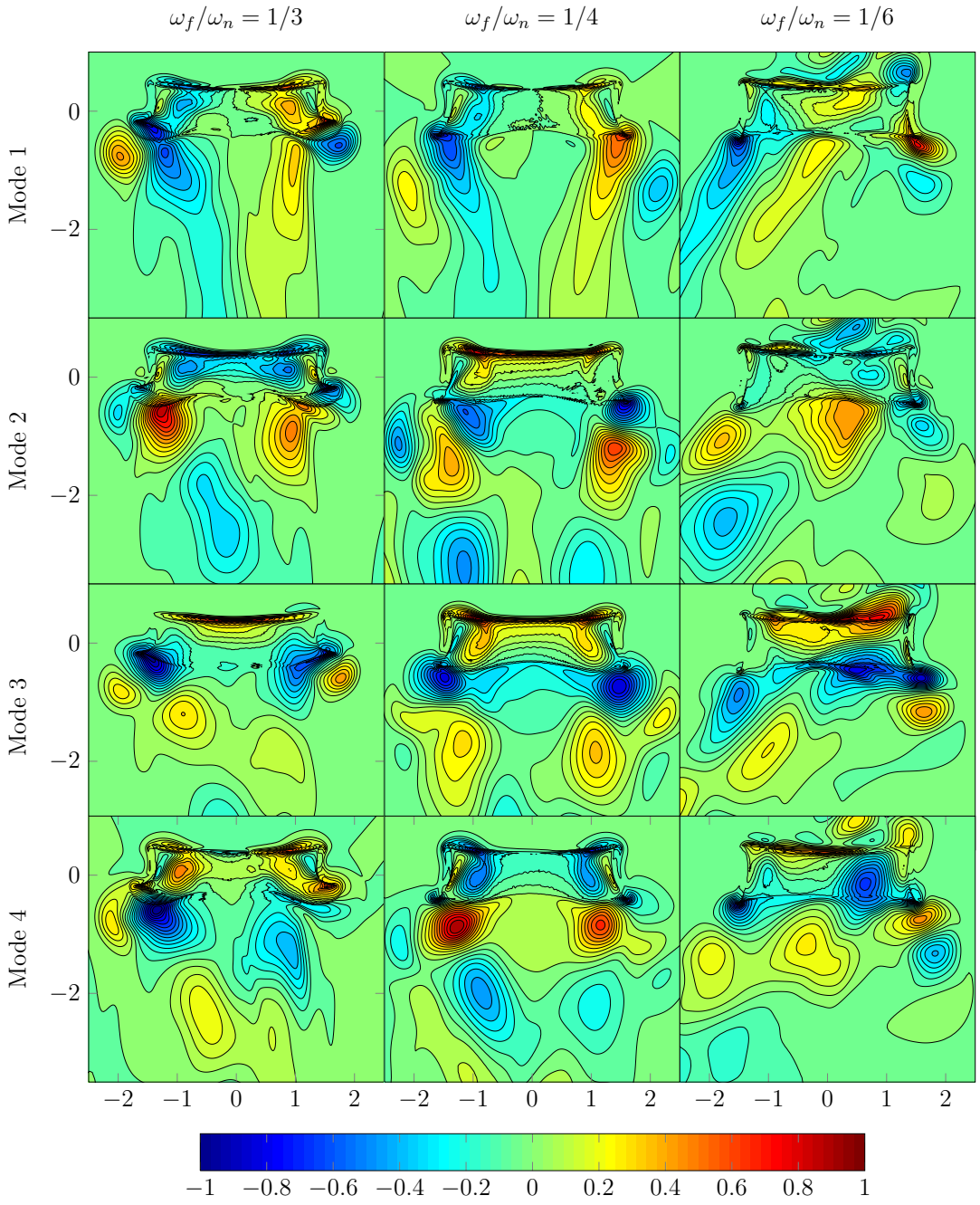


Figure 2.16: Vorticity contours determined by POD from DNS at $Re = 250$, 10 % thickness, and harmonic kinematics. Normalization is $\max_{\boldsymbol{x}} |\nabla \times \boldsymbol{\phi}_i(\boldsymbol{x})| = 1$. Horizontal scale is x/l, vertical scale is y/l.

simulation is expensive, in fact it would take several weeks for a full run of 15 periods. By contrast, the UVLM can produce 15 periods of flapping in several hours. The simple position to take is that DNS must be better since it fully models the physics of interest. However, as shown by Fitzgerald *et al.* (2011) the trends observed in the UVLM for various configurations agree well with those seen in the DNS. This provides a hybrid approach to modeling that is of use to system designers: use the UVLM as a low-fidelity prediction tool to find parametric regions of interest. Then, use DNS to compute more realistic data.

2.4.1 Flow fields

In Figs 2.17 and 2.18, comparisons among the flow fields obtained through the DNS studies for $Re = 75$, 250 and 1,000 and the UVLM studies are shown for periods 6 and 9, respectively. For both figures, the ratio of the forcing frequency to the natural frequency of the system is chosen to be $\omega_f/\omega_n = 1/2$. The magnitude of the velocity field is shown, since the vorticity field is unavailable from the UVLM simulations. In these figures, the different vortex structures can be observed and compared as the viscous diffusion in the system is decreased.

For $Re = 75$, the flow field snapshots throughout periods 6 and 9 are nearly identical, resulting in the periodicity of the flow field. This matches with the calculations of the correlation dimension, which suggests that a periodic orbit is produced in the low Reynolds number case. When $Re = 250$, the snapshots are also very similar between periods 6 and 9. However, as the correlation dimension indicates, an exact periodic solution is not seen. There are small enough differences in the flow fields to cause small disturbances to what is nearly a periodic orbit. At $Re = 1000$, with relatively low viscous diffusion, the flow field no longer appears to be periodic. However, the same kind of vortex structures can be identified as in the lower Reynolds number cases, such as the leading and trailing edge vortices. As expected,

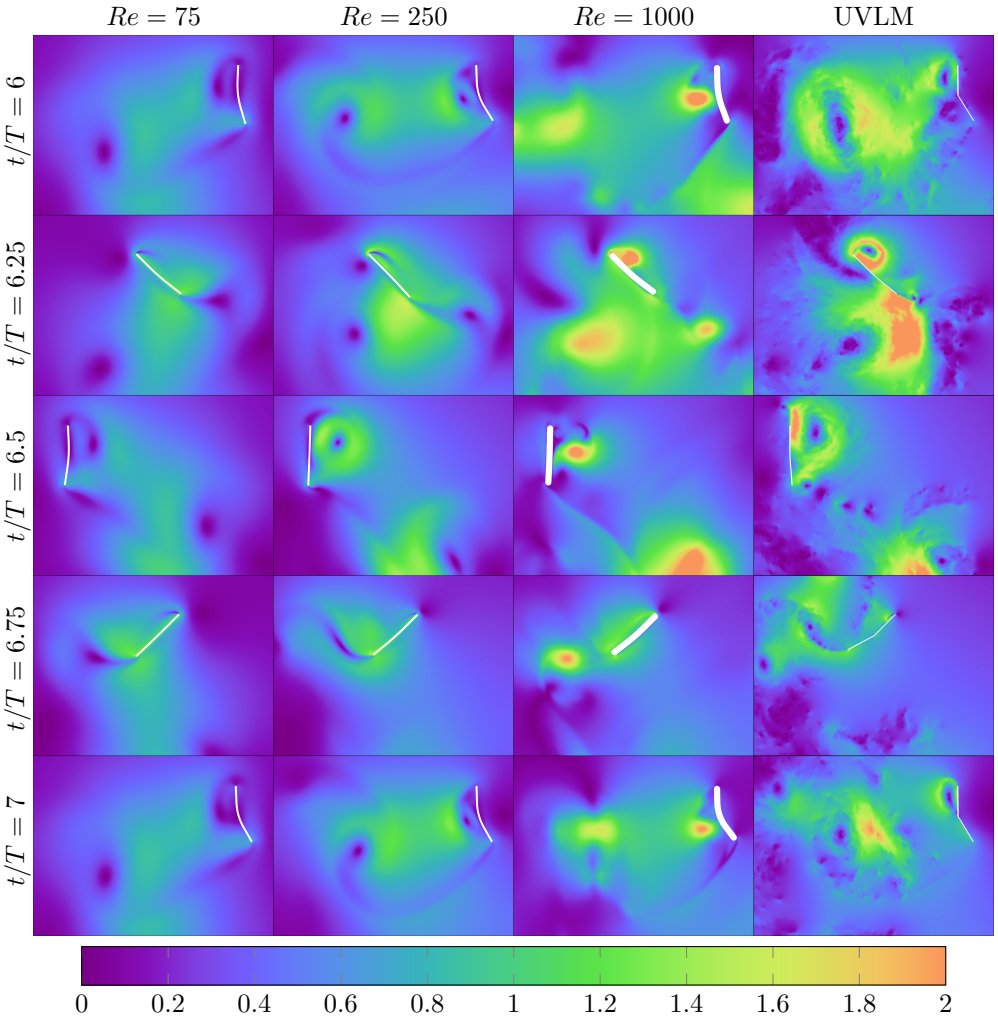


Figure 2.17: Magnitude of the velocity fields from periods 6–7 for $\omega_f/\omega_n = 1/2$, DNS at $Re \in \{75, 250, 1000\}$, and UVLM. The velocity field is normalized as $|\boldsymbol{u}|_2/u_{\text{ref}}$.

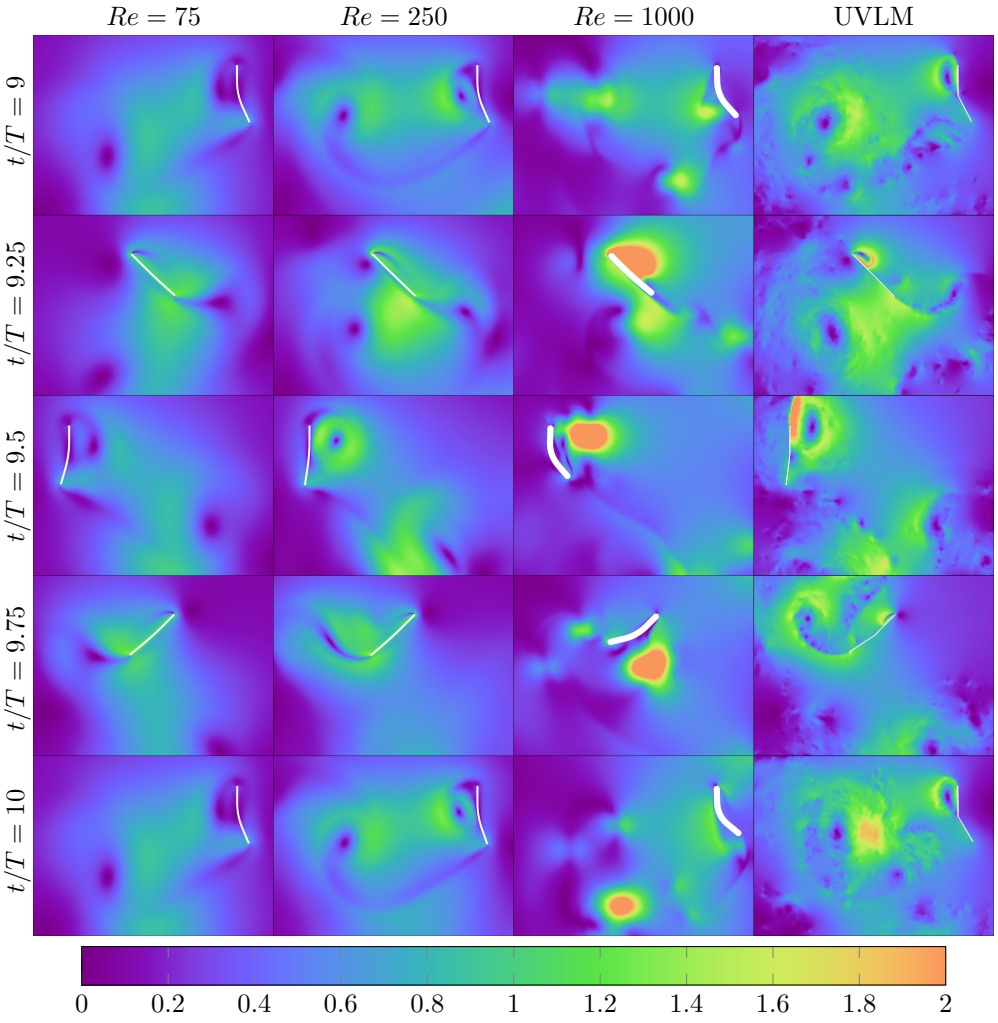


Figure 2.18: Magnitude of the velocity fields from periods 9–10 for $\omega_f/\omega_n = 1/2$, DNS at $Re \in \{75, 250, 1000\}$, and UVLM. The velocity field is normalized as $|\boldsymbol{u}|_2/u_{\text{ref}}$.

the intensity of these vortex structures increases, since they are proportional to the magnitude of the velocity. The non-periodicity of the flow field is a consequence of vortex interactions that were not dominant in the lower Reynolds numbers cases.

A sample of the the velocity POD mode shapes from the DNS and UVLM simulations are shown in Fig. 2.19. Here, the downward jet appears in the vertical velocity. The around the body also shows traces of the end of stroke vortices. The UVLM contains no viscous dissipation which is why the fluid structures do not dissipate in time. The results show that the UVLM POD is still able to nicely quantify the flow field in terms of the dominant structures.

2.4.2 Aerodynamic loads

The aerodynamic forces of interest here are the lift force and the drag. The lift is defined as the vertical force; or the force that would keep a hovering system from falling. The dimensionless lift is

$$C_L = \frac{Q_y}{\frac{1}{2}\rho_{\text{fluid}} u_{\text{ref}} l^2} \tag{2.27}$$

where Q_y is the vertical force. Since the vertical motion of the joint is zero, the drag is taken to be the horizontal force that opposes the horizontal translation. The dimensionless drag is

$$C_D = -\text{sgn}(\dot{x}) \frac{Q_x}{\frac{1}{2}\rho_{\text{fluid}} u_{\text{ref}} l^2} \tag{2.28}$$

where Q_x is the horizontal force. Observing the time series from the DNS of $Re = 1000$ and UVLM shows that initially the signals match quite well, as seen in the first period and a half in Fig. 2.20. As the simulation evolves however, the solutions appear to diverge, but still follow similar trends. The loads from the UVLM successfully capture the dominant characteristics, with similar peaks and valleys in $C_L(t)$ corresponding to

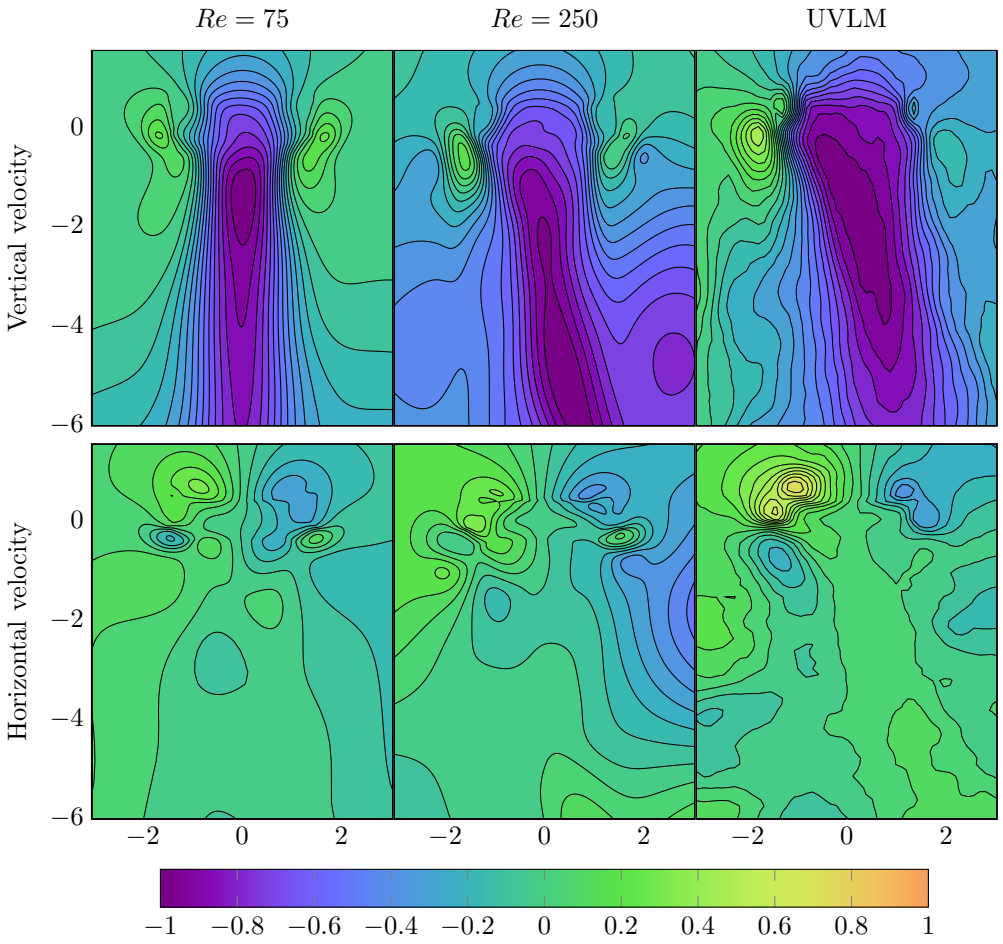


Figure 2.19: Comparisons of Mode 1 of POD velocity contours from DNS and UVLM data for $\omega_f/\omega_n = 1/3$ with harmonic kinematics. The fields are normalized by $\max_{\boldsymbol{x}} |\boldsymbol{\phi}_i(\boldsymbol{x})|_2 / u_{\text{ref}} = 1$. Horizontal scale is x/l, vertical scale is y/l.

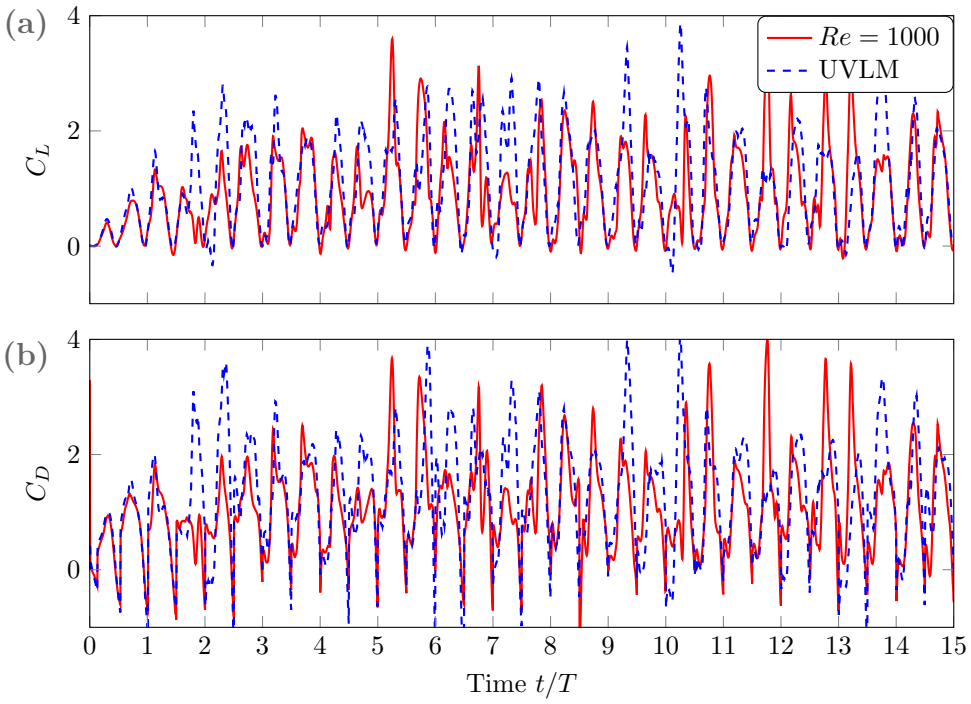


Figure 2.20: Timeseries of the dimensionless loads of the hovering profile from DNS at $Re = 1000$ --- and UVLM —— for the frequency ratio $\omega_f/\omega_n = 1/3$. (a) Lift coefficient and (b) drag coefficient.

the mid-stroke and stroke reversals, respectively. In both the lift and drag coefficients the UVLM appears to follow the same pattern of over-estimating the highs and lows. This suggests that the simulation of a transient maneuver, such as clap and fling (Weis-Fogh, 1973), and averaged long-term dynamics may be reasonably predicted using the UVLM. To explore long-term dynamics a statistical approach is needed.

Collapsing the time histories to a single period by phase-averaging is shown in Fig. 2.21. The data from periods 5 through 15 are averaged using the hovering period T as the reference clock. This range of time is used since the hovering kinematics have reached the full amplitude, and the start up transients in the fluid should have died down. At the stroke reversal, when $t/T \in \{0, 1/2\}$, there is a jump in C_D that is not fully captured by the UVLM. A possible reason for the discrepancy could be that the vortex interaction during this event is influenced more by viscosity than at other points of the cycle. For softer spring values, $\omega_f/\omega_n \in \{1/2, 1/3, 1/4\}$, the

Table 2.3: Relative error values of the phase-averaged loads between the $Re = 1000$ DNS and UVLM for a range of stiffnesses.

	Frequency Ratio ω_f/ω_n				
Rel. Error	1/2	1/3	1/4	1/6	Rigid
$\tilde{C}_D$	9.9 %	15.9 %	10.0 %	19.6 %	24.7 %
$\tilde{C}_L$	9.6 %	13.4 %	13.6 %	25.6 %	28.8 %

curves generally are in better agreement. This is particularly visible in the case of the lift. The prediction is worst for the rigid case.

The quantitative comparison of which UVLM curve best matches the corresponding DNS curve requires another definition. The error measure assumed here is a scaled L^2-norm. This is a point-wise check to see how far off the UVLM is at every point in time. This does not account for phase lag or lead between the models. For the drag force, it is written as

$$\text{error}(\tilde{C}_D) := \frac{\left(\int_0^T \left(\tilde{C}_D^{\text{DNS}}(\tau) - \tilde{C}_D^{\text{UVLM}}(\tau) \right)^2 d\tau \right)^{1/2}}{\max \tilde{C}_D^{\text{DNS}} - \min \tilde{C}_D^{\text{DNS}}} \tag{2.29}$$

with the same form being used for the lift $\tilde{C}_L$. The numerical values are compiled in Table 2.3. The differences appear to be smaller for the more compliant structures, indicating that the UVLM model would be of better use in highly flexible configurations where the structural dynamics outweigh the fluid contributions. Overall, the predictions are likely acceptable in engineering design since the compliant configuration have errors that are less than 20 %.

Taking the overall time-average of the loads provides an even better use of the UVLM. As shown in Fig. 2.22, the time averaged loads $\bar{C}_L$ and $\bar{C}_D$ trend together. The UVLM over estimates the lift and the drag, but since it is nearly by the same 15 % the ratio of the quantities match the DNS results quite well. The results indicate that moderate flexibility improves the aerodynamic performance, albeit under the

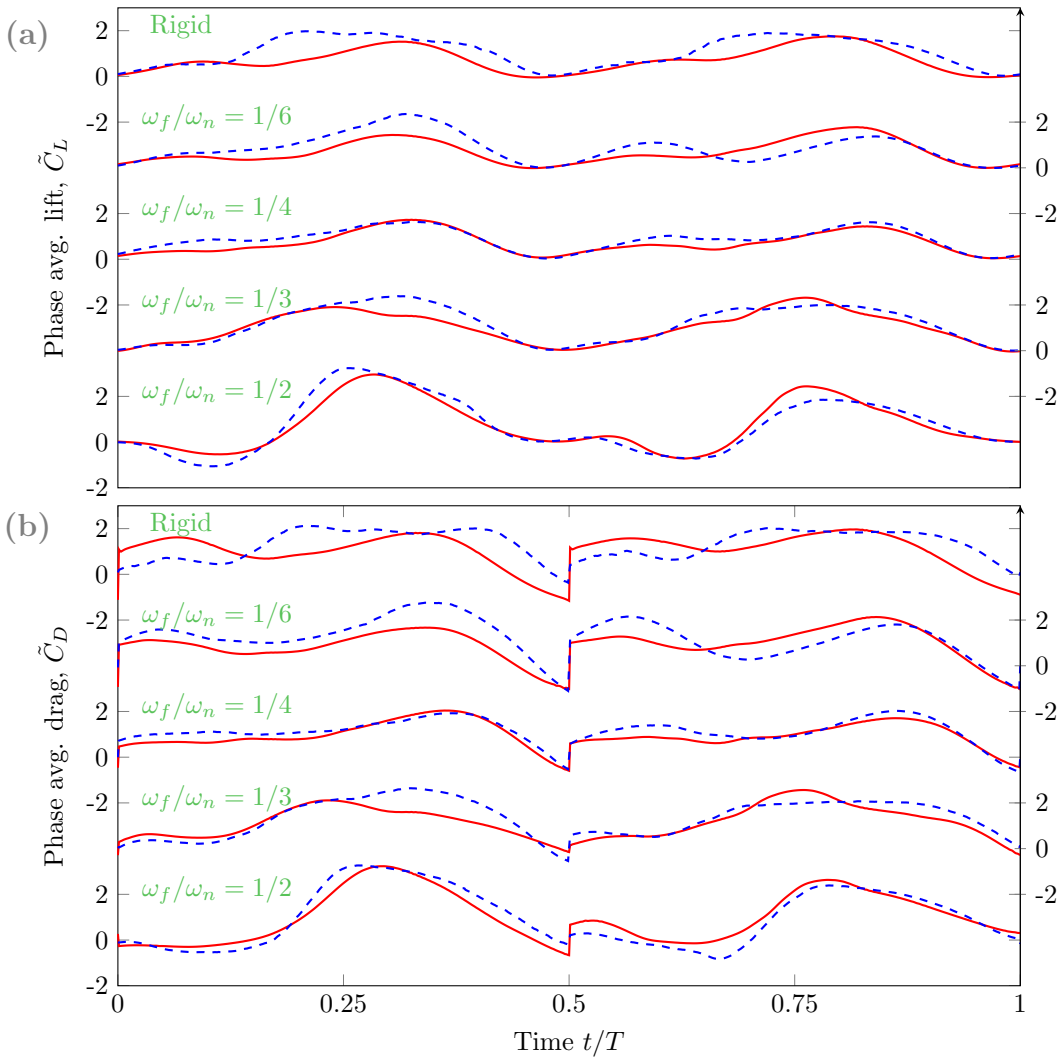


Figure 2.21: Phase averaged forces of the hovering profile from DNS at $Re = 1000$ —— and UVLM - - - for various frequency ratios. (a) Lift coefficient and (b) drag coefficient. Adapted from Fitzgerald *et al.* (2011).

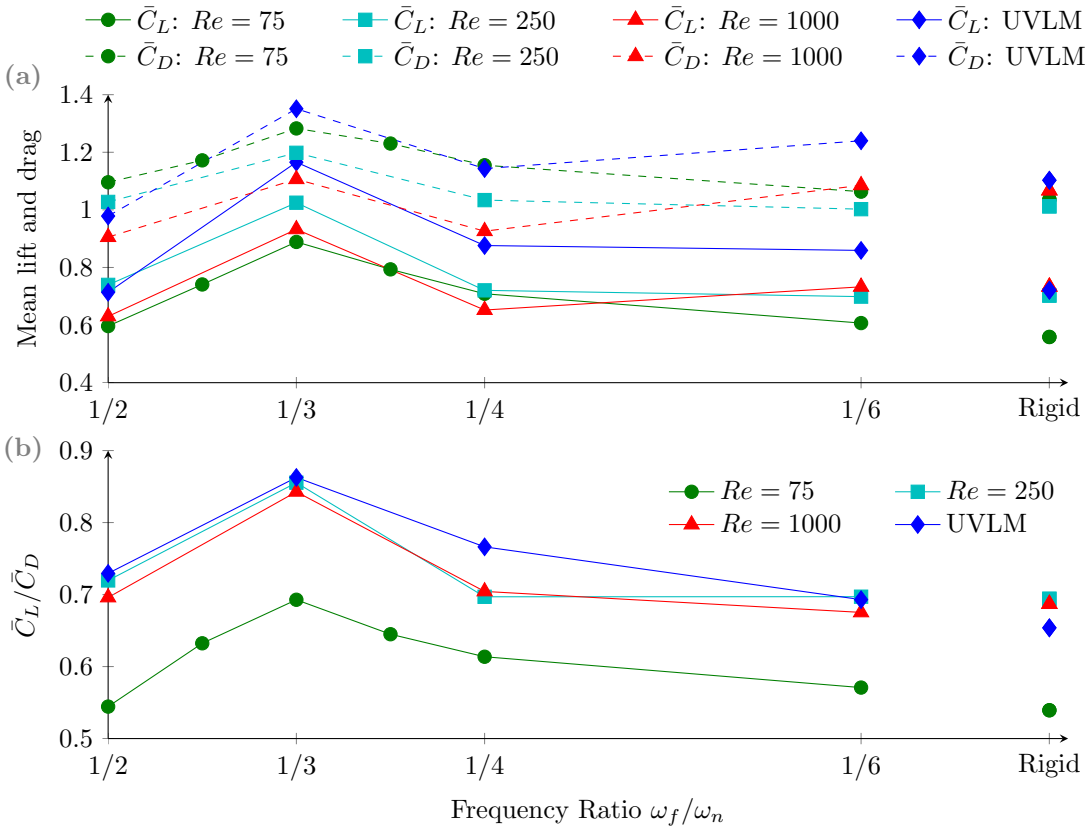


Figure 2.22: Comparisons of time averaged dimensionless lift and drag coefficients from the UVLM and DNS at various Re. (a) Mean lift and drag coefficients, (b) the ratio of mean lift to mean drag. Adapted from Fitzgerald *et al.* (2011).

modeling and kinematic assumptions.

The results of Fig. 2.21 indicate that the UVLM is adequate for quasi-steady prediction. However the results of Fig. 2.22b demonstrate that the UVLM is quite suitable for a designer to predict gross quantities and trends. The trends with respect to the spring parameter is the most significant, since it points to the UVLM being suitable for use in an optimization setting. A designer could adopt a simple hybrid approach: use the UVLM in an optimizer to find parametric regions of interest. Then investigate those regions more thoroughly with DNS.

Chapter 3

Insect wing experiments and three-dimensional structural modeling

A survey of the technology to extend the flapping work into 3D problems quickly reveals that the modeling of the structure is less well understood than the modeling of the fluid. The Navier-Stokes equation for an incompressible Newtonian fluid is an extremely versatile and well-understood model. Even a vortex lattice method in 3D is very well suited for certain problems. The body or structure, by contrast, is an open question. Not merely the geometry is variable, but the structure itself and how to model its behavior is open. Recent work on modeling the flapping wing-fluid system have either included rigid wings or highly simplified models. In Pai, Chernova, and Palazotto (2009), a body described by nonlinear structural elements interacts with a quasi-steady fluid. Experimentally, Zhao, Huang, Deng, and Sane (2010) measured the forces on a flapping wing made with Mylar. The synthesis of natural fliers to the construction of micro aerial vehicles is also a topic of open interest.

The goal of this chapter is to present a structural model that is modular enough to be adapted to a variety of geometry and material properties, and is designed to work in concert with a large-scale fluid simulation. The finite element method (FEM) provides a nice foundation of which to base the model since there is extensive literature on many aspects of its use. The FEM is also adaptable to nearly any geometry, and has the potential to be generalized to nearly any material response.

For the immersed boundary methods employed in the available large-scale fluid solver, a body with finite thickness is required. It was this same requirement that placed a surface around the structural elements in the previous 2D work. Several fluid grid points must be inside the solid body for the pressure to be resolved properly.

Therefore the body description selected was a solid body, and not a plate or shell. This provides a natural thickness to the model, and allows for future cases with highly detailed surface geometry such as a CT-scanned insect wing. A solid element also has the benefit being able to directly employ many different material laws from continuum mechanics.

This chapter begins with some modal testing performed on a living *Manduca sexta* wing. This novel experiment provides some insight and inspiration for the technical modeling that follows. In §3.2 a material model is presented that extends the classical engineering stress-strain law to finite rotations with very few assumptions. Later in §3.3 the material model is implemented as a geometrically exact finite element. In the next chapter this finite element model will be introduced to the fluid in a FSI coupling scheme.

3.1 Experiments

In the summer of 2009, during a demonstration of the Vibration Lab's new Polytec PSV-400 scanning laser vibrometer, the idea for a novel experiment was born. The concept was to measure the spectral response of an insect's wing. It was not until the experiments were underway in 2010 that the thesis of Sims (2010) was found by the author. This work masterfully examines the wings of *Manduca sexta*, and includes several studies using a Polytec vibrometer. Sims cuts off the wing to place it in a fixture attached to a shaker whereas the procedure employed here kept the insect whole and used a speaker as the excitation source. Also, Sims repeated the experiments in a vacuum chamber, which is a facility the Vibrations Lab does not current have. The findings from both studies are in good agreement.

3.1.1 Background

The bio-inspired design of flapping vehicles draws heavily from the work of both engineers and biologists. Among the various interesting aspects of insect flight is the wing itself. Largely thought to be a passively flexible structure, the details of the structure of the wing have long been an area of interest. Comstock (1918) shows a life's work of cataloging and defining the various characteristics of insect wing morphology. His naming conventions of the venation are still in common use by biologist today. In these footsteps, the works of Wootton (1992) further built on the biological map of insect wings. Early engineering-type studies to model the wing are the landmark papers of Combes and Daniel (2003a,b,c) and the PhD dissertation of Wootton's student Herbert (2001). Both groups come from a biology background, and both make claims that the flapping frequency observed is the fundamental frequency of the structure of the wing. This notation that an insect flaps at linear resonance has only recently been challenged. Computationally, this does not jive with the results shown in §2.4.2 since those models predict that efficiency decreases as linear resonance is approached (Vanella *et al.*, 2009; Fitzgerald *et al.*, 2011).

Experimentally, Sims (2010) showed that the first natural frequency of a hawkmoth wing is around twice that of the flapping frequency. The spectral information was measured via a scanning laser vibrometer from wings recently removed from living hawkmoths. The tests were repeated for several specimens in air and in a vacuum chamber, and the measured first natural frequency is around twice the flapping frequency. Kang, Aono, Cesnik, and Shyy (2011) make several scaling arguments that predict the optimal natural frequency to be around 1/3 to 1/2 of the flapping frequency. This recent work indicates that not only could older theory be wrong, but there is a tremendous opportunity to exploit nonlinear effects.

Detailed experiments focusing on the structure of the wings has been carried out by Song, Xiao, Bai, and Bai (2007) and Dirks and Taylor (2012a,b). Each of

these studies looked at minute structural details and not the wing as part of a larger coupled-system. The experimental work of Sims (2010) has been extended by O'Hara and Palazotto (2012) and new details the wings are being explored. From an FSI modeling viewpoint, the FEM solid model framework is general enough to encompass any of the material models presented in the literature, thus far.

3.1.2 Hawkmoth wing modal testing

The setup to measure the spectral response of a living insect is outlined in Fig. 3.1. A living insect is anesthetized by exposure to a large amount of Flynap[1] and then placed in a fixture molded out of modeler's clay. The forewing is fixed in place near the root with more clay and pins. The wing is acoustically excited by a JBL ASB1728 loudspeaker. This subwoofer is rated to 4000 W of continuous pink noise, and is considered high fidelity down to 20 Hz. A pseudo-random signal is output from the Polytec PSV-400 controller to a Crown MA-9000i power amplifier connected to the loudspeaker.

The choice of the *Manduca sexta* was made due to several key factors. It has been widely studied by biologist and found to be rather uniform in its body and flight characteristics across individuals. The insect wing is relatively large, and opaque. This means that it can measured by standard Polytec equipment already in the Vibrations Laboratory. This species is easily procured and grown from larvae purchased from biology supply companies. During the design of the experiment it was thought that the fundamental frequency of these insects was near 25 Hz (Combes and Daniel, 2003a,b,c). This drove the interest in an extremely low frequency speaker.

The scanning laser vibrometer is then setup to measure the response of a set of points on the surface of the wing. This mesh is shown in Fig. 3.2. The laser

[1]Flynap is a general anesthetic designed for *Musca domestica*. It is composed of 50% Triethylamine, 25% Ethanol, and 25% Fragrance, per the Material Safety Data Sheet `http://www.carolina.com/pdf/msds/FLYNAP.pdf`.

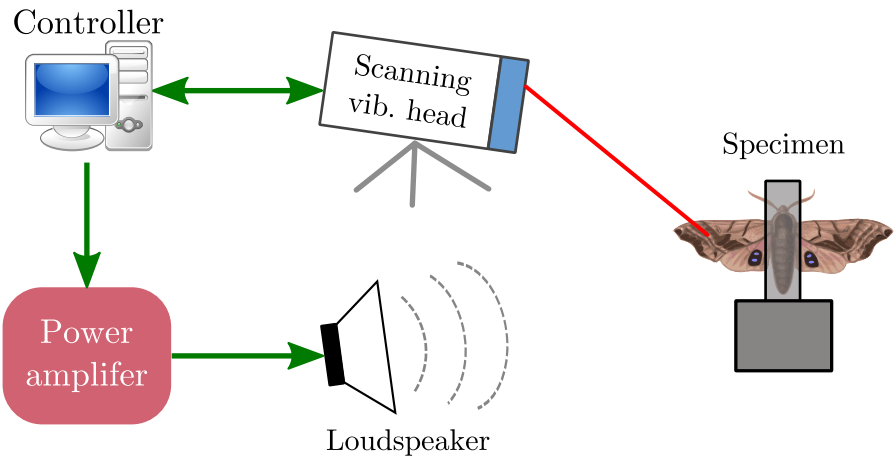


Figure 3.1: Schematic depicting the use of a scanning laser vibrometer to characterize the spectral response of a living insect wing using non-contact excitation.

vibrometer is placed such that the wing is centered and parallel in the viewfinder. The scanning is performed sequentially, and the frequency information from each point is stored as complex Fast Fourier Transform (FFT) data. The total number of FFT data points per mesh point is limited by the software, so there is a trade-off of spectral resolution when selecting the frequency range of interest. Choosing the maximum amount of FFT data at 6400 points, and selecting the frequency range of interest of 0 Hz to 1000 Hz provides a working resolution around 0.25 Hz.

Looking at the FFT data from several key points around the wing shows that first natural frequency of the wing is easily located. Figure 3.3 shows normalized FFT results from points near the root, the tip, and the trailing edge. All of these spatial points agree that the first natural frequency of that specimen is around 77 Hz. Locating the second natural frequency is a bit more challenging since the response of the wing appears rather complicated.

Searching through the visualizations of the stitched mode shapes on the PSV-400 shows that near 134 Hz there is another mode shape. The noise floor is too large at higher frequencies to be confident in locating other natural frequencies and mode shapes. This problem is inherent in the non-contact excitation using a speaker.

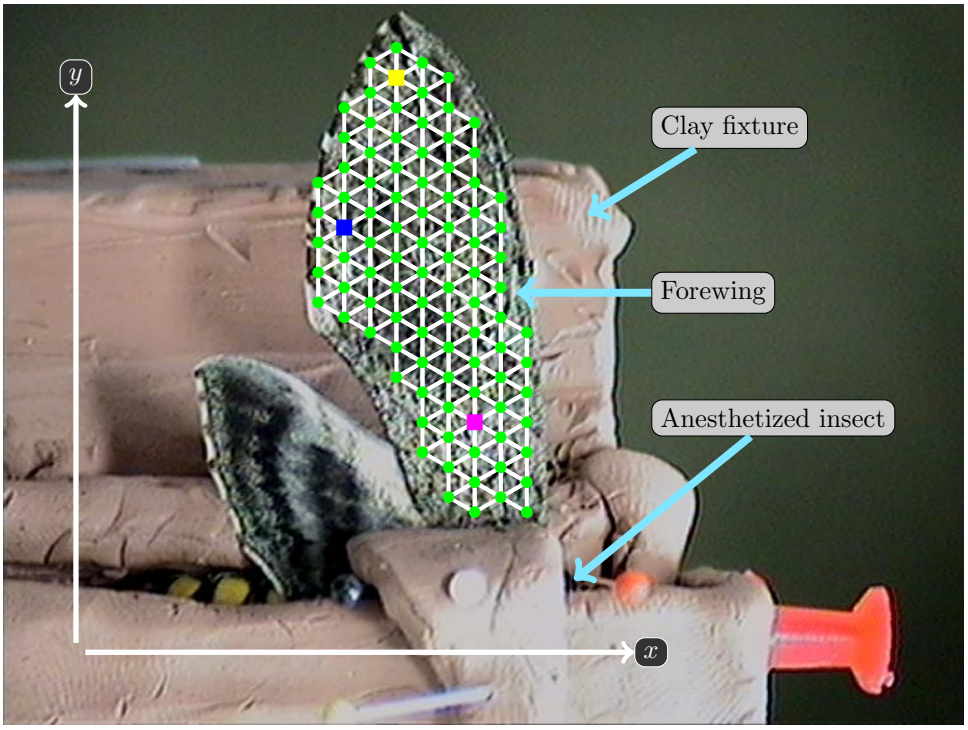


Figure 3.2: Mesh of the scanning laser vibrometer on a living *Manduca sexta* forewing. The color markers indicate locations of signal points in Fig. 3.3. Note that $+x$ is in the vertical direction.

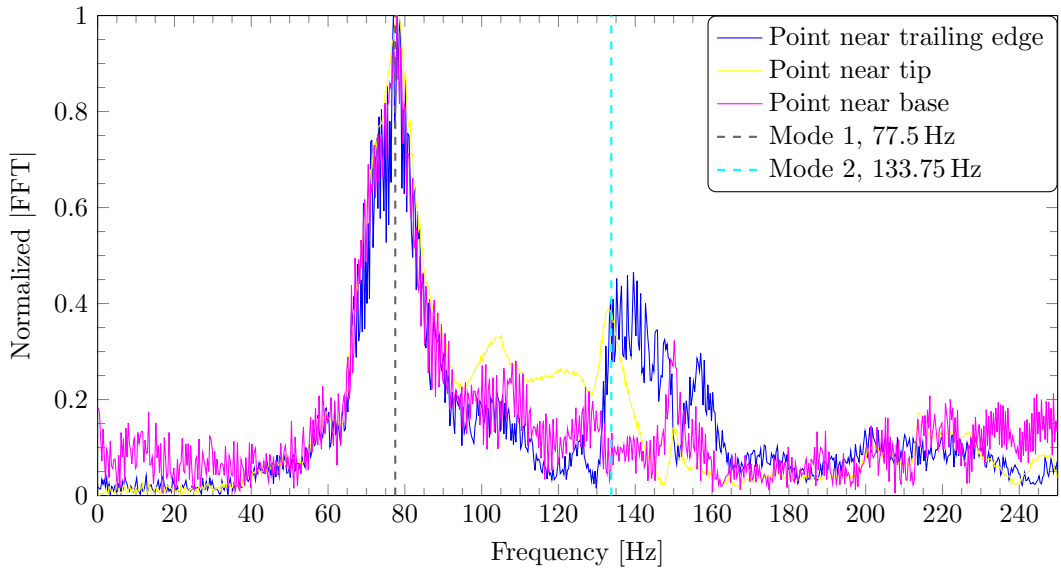


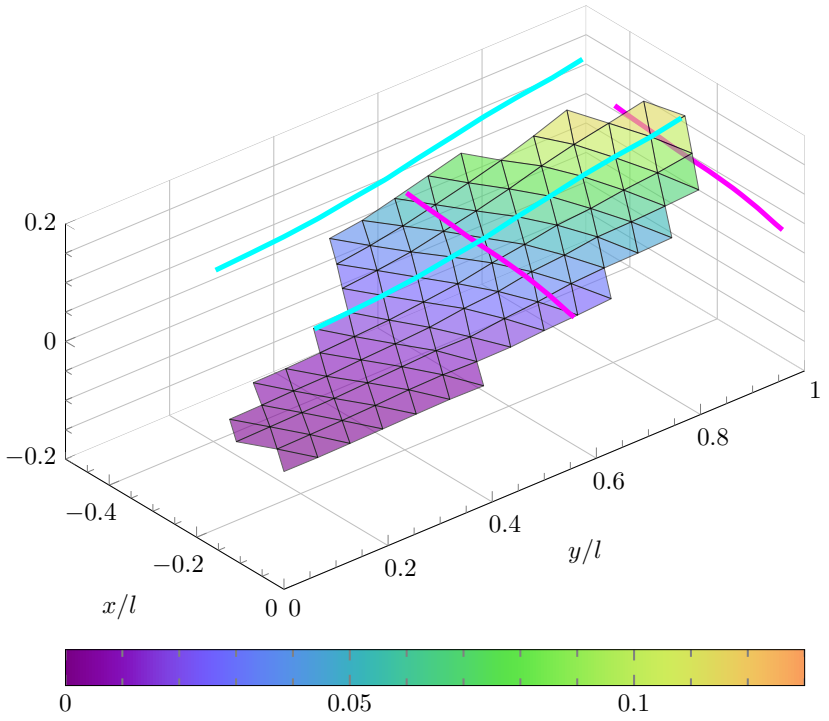
Figure 3.3: Normalized magnitude of the FFT of the velocity data determined from the laser vibrometer at various points on the forewing of a *Manduca sexta*. The color points are indicated on Fig. 3.2.

The natural frequencies reported here agree well with those of Sims (2010). He used an amputated wing directly mounted to a shaker and could therefore work with much higher frequencies and amplitudes of excitation. The novelty of the speaker experiments is that the insect and wing were alive during and after the entire measurement process. The wing measured here has not been altered by death, atrophy, or temperature. Sims had to measure the severed wings within several hours to ensure that they had not changed significantly, whereas I had around 30 minutes to ensure the insect remained asleep.

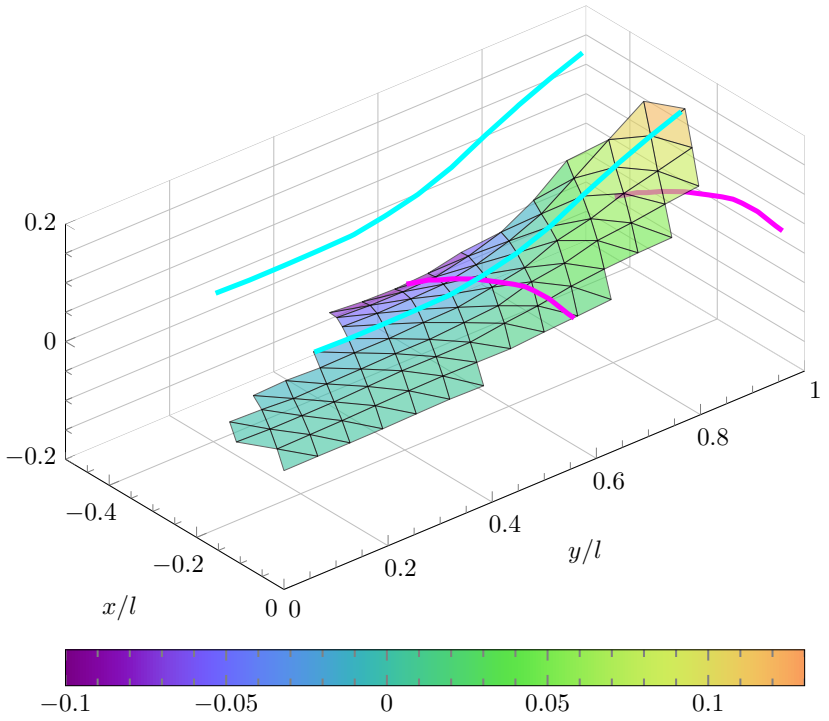
Reconstruction of the mode shapes associated with the first two natural frequencies is shown in Fig. 3.4. Here, the data was extracted from the proprietary Polytec file, and plotted in MATLAB. The x and y coordinates are scaled by the span of the wing l, and the vertical displacement of the mesh is scaled such that the maximum is $l/8$. The choice of vertical scaling is arbitrary since it represents a mode shape, and the choice of $l/8$ was chosen merely for visualization purposes. The first mode, Fig. 3.4a, appears like spanwise bending. While the second mode, Fig. 3.4b, looks like a combination of chordwise bending and some bending near the tip. A possible use for this type of detailed information is in the construction of wing models tailored to perform like a *Manduca sexta*. The distribution of material properties could be designed such that the first two natural frequencies of the model behave like the experimental results.

3.1.3 Bio-inspired wing design

Dragonflies are agile, while some butterflies can migrate thousands of miles, and each of their wings have been tailored to help the individual survive. The mode shapes of each type of insect may provide clues on how to mimic key features of that flier's traits. This natural inspiration for a wing model would provide a mechanical means to study why and how the *Manduca sexta* performs well at tasks like hovering.



(a) Mode 1 at 77.5 Hz



(b) Mode 2 at 133.75 Hz

Figure 3.4: Experimentally determined modes of a living *Manduca sexta* forewing. The x and y coordinates are scaled by the wing span, z is scaled such that the $\max z/l = 1/8$ for visualization.

The *Manduca sexta* mode shapes presented here are one part of a future library. Using the mode shapes in conjunction with the planform, the material properties of a model wing would be tailored to match the experiments. Numerically this could be performed by generating an FEM mesh, and selecting the material properties such that the first two mode shapes replicate those gather by the experiments. The specific values would be found via an optimization problem. Once a numerical study predicted property distributions of interest, a MAV designer may be interested in constructing actual wings for a vehicle.

In the present work, these experiments are used as general inspiration for the formulation. Having the ability to simulate a large variety of flexible wings, in terms of material response as well as geometry, requires a highly adaptable model. Finite element technology is a framework to build these models upon since it can be adapted to meet the modeling requirements and the implementation requirements.

3.2 Novel material law

In order to model complicated engineering structures basic assumptions about the behavior of the material must be made. There are always tradeoffs between fidelity, generality, simplicity, and how these relate to the phenomena of interest. There have been very few studies that directly measure the mechanical properties of an insect wing and virtually all these used infinitesimal engineering type tests. No material model specifically developed to predict insect wing behavior has been put forth. The naive approach is to use the classical engineering stress-strain law. This law has the well-known limitation that is cannot be used in large motion problems since the strain is not objective. Therefore an extension of the known data would be to generate an elastic material law similar to engineering stress-strain that was objective.

The simplest finite deformation law in common use is the Kirchhoff material.

In essence this law relates a not-particularly-physical stress to the square of the stretching. The result is that true stress is cubic with respect to stretch. It is theoretically simple and computationally efficient which is why it is the basis of so much practice. However the strains used in the law are not easily measured. Most interpretations call the Lagrange strains a measure of energy, not geometry. Pai and Nayfeh (1994) proposed the use of Jaumann strain and Jaumann stress tensors as the basis for an elastic law. This law was formulated for many structural elements (Pai, 2007), but not for solid elements. The goal of the present work is to extend this elastic material law to solid elements in a consistent finite element framework. The use of this type of extended engineering material law has the benefit of only requiring the two classical isotropic elastic material parameters compared to the multitude of a parameters in most hyperelastic laws. The formulation could be employed on any 3D element geometry (hexahedra, tetrahedra, etc.) as long the shape functions and suitable quadrature rules are known. The material law is also suitable for composites and anisotropic materials.

3.2.1 Background and definitions

In order to discuss material laws, some definitions and nomenclature from continuum mechanics are needed. One of the most basic quantities used is the deformation gradient tensor

$$\boldsymbol{F} := \frac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{u} + \boldsymbol{x}) = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} + \boldsymbol{I} \tag{3.1}$$

where $\boldsymbol{u}$ is displacement, $\boldsymbol{x}$ is a reference material position, and $\boldsymbol{I}$ is the identity tensor. The deformation gradient tensor is the foundation for most of solid mechanics and elasticity theory. The determinant of the deformation gradient tensor is called the Jacobian J, and it represents the volume ratio between a differential reference

volume and the deformed volume.

The polar decomposition is a unique representation of $\boldsymbol{F}$, that separates rotation and stretch under a given deformation. The rotation $\boldsymbol{R}$ and pure stretch $\boldsymbol{U}$, sometimes called the right-stretch tensor, are defined by

$$\boldsymbol{F} = \boldsymbol{R}\boldsymbol{U}. \tag{3.2}$$

The key properties of interest here are: $\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}$ and $\boldsymbol{U}$ is symmetric and positive definite (Holzapfel, 2000). There are many proposed methods in the literature to aid in the computation of $\boldsymbol{U}$ or $\boldsymbol{R}$ or both. Classically most of these methods are based on the work of Hoger and Carlson (1984); see Bouby, Fortun, Pietraszkiewicz, and Valle (2005) for a more recent list of methods. The computation of $\boldsymbol{U}$ for example boils down to the tensor square root of the Right Cauchy-Green deformation tensor $\boldsymbol{C} = \boldsymbol{F}^T\boldsymbol{F} = \boldsymbol{U}^2$. The simplest way to numerically compute $\boldsymbol{U}$ in a Cartesian frame is by the singular value decomposition of $\boldsymbol{F}$. Such an operation on a 3×3 matrix is not terribly expensive and computationally robust methods are freely found, such as `DGESVD` in LAPACK (Anderson *et al.*, 1999). Using the singular value decomposition gives

$$\boldsymbol{F} = \boldsymbol{\Psi}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T, \tag{3.3}$$

which provides for the direct construction of

$$\boldsymbol{U} = \boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T \tag{3.4a}$$

$$\boldsymbol{R} = \boldsymbol{\Psi}\boldsymbol{\Lambda}^T. \tag{3.4b}$$

This particular construction also has the added benefit of providing the spectral decomposition of $\boldsymbol{U}$ without any additional labor. The principal values of $\boldsymbol{U}$ are the

principal stretches $\{\lambda_1, \lambda_2, \lambda_3\}$, and are also the values of the diagonal $\boldsymbol{\Sigma}$. This will be exploited later when discussing the hyperelastic potential function.

There are many ways to objectively measure strain, and each may have a work conjugate measure of stress, the most commonly used measures are the Seth-Hill stress-strain tensors (Farahani and Naghdabadi, 2003). For strain measures defined in the reference configuration, these can be written as

$$\bar{\boldsymbol{E}}(\boldsymbol{U};\, m) = \begin{cases} \ln(\boldsymbol{U}), & m = 0 \\ \dfrac{1}{m}\left(\boldsymbol{U}^m - \boldsymbol{I}\right), & m \neq 0 \end{cases} \tag{3.5}$$

where m need not be an integer in general. All the classical measures of strain are defined in (3.5). Some examples are the logarithmic Hencky strain ($m = 0$), or the Green-Lagrange strain ($m = 2$). The Green-Lagrange strain, denoted after this simply as $\boldsymbol{E}$, is used widely throughout the literature. It is computationally simple to build since it exploits the quadratic form of $\boldsymbol{U}^2 = \boldsymbol{F}^T\boldsymbol{F}$ and therefore $\boldsymbol{U}$ is not computed explicitly. Particular interest will be placed on the Biot, or sometimes Jaumann, strain $m = 1$. Here the Biot strain will be referred to as

$$\boldsymbol{L} = \boldsymbol{U} - \boldsymbol{I}. \tag{3.6}$$

It is important to note that each of these strain measures can be transformed into the others. Therefore a particular elastic law involving one strain measure can be transformed into using other measures, but still be the same law.

The ways to measure stress are as varied as the ways to measure strain. Each defines a force per unit area with regards to different frames, configurations, or push-pull operators. The stresses of interest herein are the Cauchy (true) stress $\boldsymbol{\sigma}$, the 2nd Piola-Kirchhoff stress $\boldsymbol{S}$, and the symmetric Biot stress $\boldsymbol{G}$. It is worth pointing out that the knowledge of any one of these stress measures, and $\boldsymbol{F}$ is enough

to compute any of the other stress measures. For example, the relation between the 2nd Piola-Kirchhoff stress and the Biot stress is the well-known relation (Ogden, 1984)

$$\boldsymbol{G} = \frac{1}{2}\left(\boldsymbol{U}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{U}\right). \tag{3.7}$$

If $\boldsymbol{U}$ and $\boldsymbol{G}$ are known, then in a Cartesian setting $\mathbf{S}$ can be viewed as the solution to a $\mathbb{R}^{3\times 3}$ linear Lyapunov equation. Jameson (1968) provides a direct solution for 3×3 matrices in this type of problem.

The tensors $\boldsymbol{G}$ and $\boldsymbol{L}$ are work conjugate, and therefore the virtual internal work can be expressed as

$$\delta W_{\text{int}} = \int_{\Omega_0} \boldsymbol{G} : \delta \boldsymbol{L} \; dV_0. \tag{3.8}$$

The majority of the technical effort of Pai (2007) is devoted to the construction of $\boldsymbol{L}$ and $\delta\boldsymbol{L}$ for various structural elements like beams and plates. This is complicated since $\boldsymbol{L}$ involves the polar decomposition. Since any work-conjugate pair can be used to maintain geometric exactness, then other simpler forms of δW_{int} can be used here. This form does not specify the material law, just the geometric description. If instead, the standard Green-Lagrange form of the virtual work is used

$$\delta W_{\text{int}} = \int_{\Omega_0} \boldsymbol{S} : \delta \boldsymbol{E} \; dV_0 \tag{3.9}$$

an application of the Biot law merely fits inside it with some transformations. An excellent reference for developing finite elements based on (3.9) is Belytschko, Liu, and Moran (2000).

In an incremental, or iterative, method the virtual internal work is a function of the degrees of freedom (DOF) of the body $\mathbf{q}$. The iterative process is usually

introduced to solve a Newton-Raphson style problem to find $\mathbf{q}$ for equilibrium. Expanding the virtual internal work provides a representation of the current internal forces $\mathbf{f}_{\text{int}}$, and the derivative of these forces with respect to the DOF. Looking at this expansion gives the following definitions.

$$\delta W_{\text{int}} = \int_{\Omega_0} \boldsymbol{S} : \delta \boldsymbol{E} \; dV_0 = \delta \mathbf{q}^T \mathbf{f}_{\text{int}} \tag{3.10}$$

$$= \delta \mathbf{q}^T \mathbf{f}^0_{\text{int}} + \delta \mathbf{q}^T \mathbf{K} \left(\mathbf{q} - \mathbf{q}_0 \right) + \cdots \tag{3.11}$$

Where the internal forces are computed as

$$\mathbf{f}_{\text{int}} = \int_{\Omega_0} \mathbf{B}^T \{\mathbf{S}\} \; dV_0 \, . \tag{3.12}$$

Here $\mathbf{B}$ is the usual matrix of derivatives from the variation of $\boldsymbol{E}$ (Belytschko *et al.*, 2000, e.g. §4.9.2), and $\{\mathbf{S}\}$ is the current $\boldsymbol{S}$ in Voigt notation; see A.3.2 for the technical details. Similarly, the stiffness matrix $\mathbf{K}$ is also built. The stiffness can be decomposed into the material and the geometric contributions.

$$\mathbf{K} := \mathbf{K}_{\text{mat}} + \mathbf{K}_{\text{geo}} \tag{3.13a}$$

$$\mathbf{K}_{\text{mat}} := \int_{\Omega_0} \mathbf{B}^T \frac{\partial \{\mathbf{S}\}}{\partial \mathbf{q}} \; dV_0 = \int_{\Omega_0} \mathbf{B}^T \mathbf{C}^{SE} \mathbf{B} \; dV_0 \tag{3.13b}$$

The geometric stiffness is a function of $\boldsymbol{S}$ and the reference geometry. It is not a function of the derivatives of $\boldsymbol{S}$, again the appendix contains details. Different material models are implemented by changing how $\{\mathbf{S}\}$ is computed. Some FEM codes rely on the second elasticity tensor $\mathbf{C}^{SE}$, while others directly compute $\frac{\partial \{\mathbf{S}\}}{\partial \mathbf{q}}$. There are some special cases in commercial FEM codes that use the hyperelastic potential function directly. The use of the potential function greatly simplifies the amount of coding needed in these commercial codes compared to general user-defined

material models.

Finite elements are used throughout to approximate the displacement field of the body. The DOF are assumed to be displacements $\{u_1, u_2, u_3\} \in \mathbb{E}^3$, and are approximated by spatial polynomials of compact support over each element.

$$\boldsymbol{u}(\boldsymbol{x}, t) \approx \mathbf{u} = \mathbf{N}(\boldsymbol{\xi})\mathbf{q}(t) \tag{3.14}$$

Here, $\boldsymbol{\xi} = (\xi, \eta, \zeta)$ is the natural coordinates of the volume element. Each element has n_{ee} DOF, and the body is made up of n_{el} elements. The mesh is composed of n_{np} nodes, and the body has a total of n_{dof} DOF.

3.2.2 Biot material

In linear mechanics, all the strain (stress) measures are the same, but this is not true for finite deformation. The linear stress-strain law, for a possibly anisotropic material, can be written as

$$\boldsymbol{\sigma} = \mathfrak{C} : \boldsymbol{\varepsilon} \tag{3.15}$$

where $\boldsymbol{\sigma}$ is the stress, $\boldsymbol{\varepsilon}$ is the strain, and $\mathfrak{C}$ is the 4th order tensor of material constants, with the usual symmetries. In a uni-dimensional scenario this equation becomes

$$\sigma = E\frac{du}{dx} = E(\lambda - 1) \tag{3.16}$$

where E is Young's Modulus. The derivative can be related to the local stretch as $\frac{du}{dx} = \lambda - 1$. The Kirchhoff material model simply replaces the stresses and strains of

this law with objective measures and retains the same set of material constants.

$$\boldsymbol{S} = \mathfrak{C} : \boldsymbol{E} \tag{3.17}$$

This model is no longer linear with regards to stretch, it is cubic. There are severe issues with this model: in compression it softens nonphysically (Holzapfel, 2000), and in tension it stiffens. Recently Pai and Nayfeh (1994); Pai and Palazotto (1995) proposed a law were the Biot stress $\boldsymbol{G}$ and strain $\boldsymbol{L}$ are used in place of $\boldsymbol{S}$ and $\boldsymbol{E}$. The analysis presented by Pai and coworkers refers to the Jaumann stress and strain which are more commonly called the Biot stress and strain. Also their formulations are vectorial which become very lengthy. The presentation here will be using tensors, both for compactness and generality. The Biot material model is simply

$$\boldsymbol{G} = \mathfrak{C} : \boldsymbol{L}. \tag{3.18}$$

In a uni-dimensional scenario $\boldsymbol{L} \to \lambda - 1$, and the stress-strain relationship collapses back to the engineering law.

The justification for the choice of this empirical law is that it is theoretically simpler than the Kirchhoff material law. The Biot model in the absence of rotations is identical to the engineering stress-strain law. The advantage over the classical linear model is the ability to work over arbitrary rigid body motions. The applicability of the law is similar to Kirchhoff, large motion but small to moderate strains. An alternative point of view is to compare the material model to corotational finite elements (Felippa and Haugen, 2005). In most corotational finite elements, a local coordinate system is reconstructed for each element that accounts for the rigid body motion of the element, and then strains are measured with respect to this convecting frame. A similar concept is being used in the Biot material. At each material point of the element a pure stretch is being computed. Theoretically this allows for the

element sizing to be much larger since the element size is not related to the body's ability to represent rigid motion.

There are many possible ways to formulate a material law for use with finite elements. In §3.2.3 the law is formulated as an isotropic hyperelastic strain potential. This formulation is useful for integrating the material model into existing FEM implementations including commercial products like ANSYS$^{\text{TM}}$. In §3.2.4 the Biot model is constructed and linearized in a more general finite element setting.

3.2.3 Hyperelasticity law

The Biot material law is a hyperelastic law, and therefore the strain potential is an exact differential. Knowing this density function in closed analytic form is useful for custom material models in commercial finite-element products. For example, the ANSYS user defined subroutine `UserHyper` provides a convenient way to define any isotropic hyperelastic law in terms of the three scaled invariants of the right Cauchy-Green deformation tensor (ANSYS, 2009). There are also more complicated routines for anisotropic material models. In ANSYS the user material response is called `UserMat` while in Abaqus$^{\text{TM}}$ it is called `UMAT`/`VUMAT` (ABAQUS, 2012).

The isotropic form of the law requires two constants and can be written as

$$\boldsymbol{G} = \mathfrak{C} : \boldsymbol{L} = \gamma_1 \mathrm{Tr}\,(\boldsymbol{L})\,\boldsymbol{I} + \gamma_2 \boldsymbol{L}. \tag{3.19}$$

Where γ_1 is Lamé's first parameter and γ_2 is twice the shear modulus. Put in terms of the Young's modulus E and Poisson's ratio ν gives

$$\gamma_1 = \frac{E\nu}{(1+\nu)(1-2\nu)} \tag{3.20a}$$

$$\gamma_2 = 2\frac{E}{2(1+\nu)}. \tag{3.20b}$$

This form of $\mathfrak{C}$ provides many simplifications, both for computing $\boldsymbol{S}$ as well as its linearization. The stretch tensor $\boldsymbol{U}$ and the stress $\boldsymbol{G}$ are coaxial since the law is isotropic. It can be shown that this implies that $\boldsymbol{U}$ and $\boldsymbol{S}$ are also coaxial, and that (3.7) can be simplified for the isotropic case, see §A.4.1 for details.

3.2.3.1 Potential function

In principal coordinates, the stretch tensor $\boldsymbol{U} \to \boldsymbol{\Sigma}$ is diagonal in Cartesian coordinates, with the stretches λ_a on the diagonal. The second Piola-Kirchhoff stress becomes

$$\begin{aligned}\breve{\mathbf{S}} &= \breve{\mathbf{U}}^{-1}\breve{\mathbf{G}} = \breve{\mathbf{U}}^{-1}\left(\gamma_1 \mathrm{Tr}(\breve{\mathbf{U}}-\mathbf{I})\mathbf{I} + \gamma_2(\breve{\mathbf{U}}-\mathbf{I})\right) \\ &= \begin{bmatrix} \frac{\gamma_2(\lambda_1-1)+\gamma_1(\lambda_1+\lambda_2+\lambda_3-3)}{\lambda_1} & 0 & 0 \\ 0 & \frac{\gamma_2(\lambda_2-1)+\gamma_1(\lambda_1+\lambda_2+\lambda_3-3)}{\lambda_2} & 0 \\ 0 & 0 & \frac{\gamma_2(\lambda_3-1)+\gamma_1(\lambda_1+\lambda_2+\lambda_3-3)}{\lambda_3} \end{bmatrix}.\end{aligned}$$

This can be written more compactly as

$$S_a = \frac{1}{\lambda_a}\left(\gamma_2(\lambda_a - 1) + \gamma_1(\lambda_1+\lambda_2+\lambda_3-3)\right), \text{ for } a = 1,2,3. \tag{3.21}$$

Using the well-known relation between the potential Ψ and S_a (Holzapfel, 2000)

$$S_a = \frac{1}{\lambda_a}\frac{\partial \Psi}{\partial \lambda_a} \tag{3.22}$$

the density function can be constructed since it is an exact differential. Beginning with $a = 1$

$$\begin{aligned}\int S_1\lambda_1 \, d\lambda_1 &= \int \gamma_2(\lambda_1 - 1) + \gamma_1(\lambda_1+\lambda_2+\lambda_3-3)\, d\lambda_1 \\ &= \frac{\gamma_1\lambda_1^2}{2} - 3\gamma_1\lambda_1 + \gamma_1\lambda_2\lambda_1 + \gamma_1\lambda_3\lambda_1 + \frac{\gamma_2\lambda_1^2}{2} - \gamma_2\lambda_1 + f_1(\lambda_2,\lambda_3)\end{aligned}$$

Now, taking the derivative of this expression with respect to λ_2 gives a relationship to S_2.

$$\frac{1}{\lambda_2}\frac{\partial}{\partial\lambda_2}\left(\frac{\gamma_1\lambda_1^2}{2} - 3\gamma_1\lambda_1 + \gamma_1\lambda_2\lambda_1 + \gamma_1\lambda_3\lambda_1 + \frac{\gamma_2\lambda_1^2}{2} - \gamma_2\lambda_1 + f_1\left(\lambda_2, \lambda_3\right)\right) = S_2.$$

Therefore,

$$\begin{aligned}\int \frac{\partial}{\partial\lambda_2} f_1(\lambda_2, \lambda_3)\ d\lambda_2 &= \int \gamma_1\lambda_2 + \gamma_1\lambda_3 - 3\gamma_1 + \gamma_2\lambda_2 - \gamma_2\ d\lambda_2 \\ &= \frac{\gamma_1\lambda_2^2}{2} - 3\gamma_1\lambda_2 + \gamma_1\lambda_3\lambda_2 + \frac{\gamma_2\lambda_2^2}{2} - \gamma_2\lambda_2 + f_2(\lambda_3).\end{aligned}$$

Putting this back into the previous results and repeating the same procedure for $a = 3$ gives

$$\begin{aligned}\frac{1}{\lambda_3}\frac{\partial}{\partial\lambda_3}\Bigg(&\frac{\gamma_1\lambda_1^2}{2} - 3\gamma_1\lambda_1 + \gamma_1\lambda_2\lambda_1 + \gamma_1\lambda_3\lambda_1 + \frac{\gamma_2\lambda_1^2}{2} - \gamma_2\lambda_1 \\ &+\frac{\gamma_1\lambda_2^2}{2} - 3\gamma_1\lambda_2 + \gamma_1\lambda_3\lambda_2 + \frac{\gamma_2\lambda_2^2}{2} - \gamma_2\lambda_2 + f_2(\lambda_3)\Bigg) = S_3\end{aligned}$$

Integrating one final time yields

$$\begin{aligned}\int \frac{\partial}{\partial\lambda_3} f_2(\lambda_3)\ d\lambda_3 &= \int \gamma_1\lambda_3 - 3\gamma_1 + \gamma_2\lambda_3 - \gamma_2\ d\lambda_3 \\ &= \frac{\gamma_1\lambda_3^2}{2} - 3\gamma_1\lambda_3 + \frac{\gamma_2\lambda_3^2}{2} - \gamma_2\lambda_3 + c.\end{aligned}$$

Now the potential, unique to a constant, can be written as

$$\begin{aligned}\Psi(\lambda_1, \lambda_2, \lambda_3) = \frac{1}{2}(\gamma_1 + \gamma_2)\left(\lambda_1^2 + \lambda_2^2 + \lambda_3^2\right) + \gamma_1\left(\lambda_1\lambda_2 + \lambda_3\lambda_2 + \lambda_1\lambda_3\right) \\ - \left(3\gamma_1 + \gamma_2\right)\left(\lambda_1 + \lambda_2 + \lambda_3\right) + \left(\frac{9\gamma_1}{2} + \frac{3\gamma_2}{2}\right) \quad (3.23)\end{aligned}$$

To set the constant, $\Psi(1,1,1) = 0$ is usually preferred, therefore $c = \frac{9\gamma_1}{2} + \frac{3\gamma_2}{2}$. Appendix A.4.2 contains the proof that this model is consistent with classical theory.

In order to use the routines like `UserHyper`, the potential function and its derivatives must be in terms of the scaled invariants of $\boldsymbol{C}$. Namely in terms of

$$J = \sqrt{III_C} = III_U = \lambda_1\lambda_2\lambda_3 \tag{3.24a}$$

$$\bar{I}_C = J^{-2/3} I_C = J^{-2/3}\left(\lambda_1^2 + \lambda_2^2 + \lambda_3^2\right) \tag{3.24b}$$

$$\bar{II}_C = J^{-4/3} II_C = J^{-4/3}\left(\lambda_1^2\lambda_2^2 + \lambda_1^2\lambda_3^2 + \lambda_2^2\lambda_3^2\right) \tag{3.24c}$$

However the potential does not readily admit a realization in terms of these invariants. Using I_U and II_U as the first and second invariants of $\boldsymbol{U}$ Ψ is restated as

$$\Psi = \frac{1}{2}(\gamma_1 + \gamma_2)J^{2/3}\bar{I}_C + \gamma_1 II_U - (3\gamma_1 + \gamma_2)I_U + \left(\frac{9\gamma_1}{2} + \frac{3\gamma_2}{2}\right) \tag{3.25}$$

The closed form of I_U and II_U as functions of $(\bar{I}_C, \bar{II}_C, J)$ is quite involved and requires the roots of cubic and quartic equations. In one realization of the solution, developed by Norris (2007), the closed form set of functions are

$$\zeta(x,y) = \frac{27 + 2x^3 - 9xy}{2(x^2 - 3y)^{3/2}} \tag{3.26a}$$

$$g(x,y) = \sqrt{1/3}\left(x + \sqrt{x^2 - 3y}\left\{\left(\zeta(x,y) + \sqrt{\zeta(x,y)^2 - 1}\right)^{1/3} + \left(\zeta(x,y) - \sqrt{\zeta(x,y)^2 - 1}\right)^{1/3}\right\}\right)^{1/2} \tag{3.26b}$$

$$\phi(x,y) = g(x,y) + \sqrt{x - g(x,y)^2 + \frac{2}{g(x,y)}} \tag{3.26c}$$

$$I_U(\bar{I}_C, \bar{II}_C, J) = J^{1/3}\phi(\bar{I}_C, \bar{II}_C) \tag{3.26d}$$

$$II_U(\bar{I}_C, \bar{II}_C, J) = J^{2/3}\phi(\bar{II}_C, \bar{I}_C). \tag{3.26e}$$

Since this expression is explicit, its various derivatives with respect to the scaled invariants can be computed. However, due to the construction of the function, this is not numerically robust in finite arithmetic. Further, it is important to point out that while $g(x,y)$ is always real certain sub expressions in the calculation may be complex.

In a strain-free configuration, $J = 1, \bar{I}_C = 3, \bar{II}_C = 3$. This poses a problem for the denominator of ζ. Carefully evaluating the limit gives $\zeta \to 0$, $\Psi \to 0$, but numerically this becomes undefined. This makes an implementation, such as one in Fortran, a little more complicated than it would at first seem but is still straightforward. A special case for equilibrium must be included to set $\zeta \to 0$.

To complete the implementation, the first and second derivatives must also be computed. Using ANSYS as an example, the derivatives needed are the gradient and Hessian $\{\frac{\partial\Psi}{\partial\bar{I}_C}, \frac{\partial\Psi}{\partial\bar{II}_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial\bar{I}_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial\bar{II}_C}, \frac{\partial^2\Psi}{\partial\bar{II}_C\partial\bar{II}_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial J}, \frac{\partial^2\Psi}{\partial\bar{II}_C\partial J}, \frac{\partial\Psi}{\partial J}, \frac{\partial^2\Psi}{\partial J\partial J}\}$. A robust method to compute these are to employ the basic identities relating the invariants. Using this method the derivatives of I_C and II_C with respect to the scaled invariants can be used to construct the needed derivatives of Ψ. Implicit formulas can be constructed for the various first derivatives, and then second derivatives. These formulae are presented in §A.4.3. A sample code is provided in §B.1 for using a Biot material in ANSYS.

3.2.3.2 Second elasticity tensor

Another common method commonly employed in the linearization of finite-element formulations is the rate form. Here, the second Piola-Kirchhoff stress is related to the rate of the Green-Lagrange strain as

$$\dot{\boldsymbol{S}} = \mathfrak{C}^{SE} : \dot{\boldsymbol{E}}$$

where $\mathfrak{C}^{SE}$ is the so called second elasticity tensor

$$\mathfrak{C}^{SE} := 4\frac{\partial^2 \Psi}{\partial \boldsymbol{C} \partial \boldsymbol{C}}. \tag{3.27}$$

The use of $\mathfrak{C}^{SE}$ comes into play for the material portion of the tangent stiffness matrix. For an isotropic hyperelastic material, this tensor can be written in closed form in terms of principal directions and stretches (Holzapfel, 2000).

$$\mathfrak{C}^{SE} = \sum_{a,b=1}^{3} \frac{1}{\lambda_b}\frac{\partial S_a}{\partial \lambda_b} \hat{\boldsymbol{n}}_a \otimes \hat{\boldsymbol{n}}_a \otimes \hat{\boldsymbol{n}}_b \otimes \hat{\boldsymbol{n}}_b + \sum_{\substack{a,b=1 \\ a \neq b}}^{3} \frac{S_b - S_a}{\lambda_b^2 - \lambda_a^2} \left(\hat{\boldsymbol{n}}_a \otimes \hat{\boldsymbol{n}}_b \otimes \hat{\boldsymbol{n}}_a \otimes \hat{\boldsymbol{n}}_b + \hat{\boldsymbol{n}}_a \otimes \hat{\boldsymbol{n}}_b \otimes \hat{\boldsymbol{n}}_b \otimes \hat{\boldsymbol{n}}_a \right)$$

Here S_a is the principal 2nd Piola-Kirchhoff stress, and the unit vector $\hat{\boldsymbol{n}}_a$ can be viewed as the a-th column of $\boldsymbol{\Lambda}$ in matrix form. Using this equation the 81 components of $\mathfrak{C}^{SE}$ could be computed. This is not necessary since the elasticity tensor is symmetric. This tensor can be realized as a 6×6 symmetric matrix using

Voigt notation.

$$\mathfrak{C}^{SE} \rightarrow \mathbf{C}^{SE} = \begin{bmatrix} c_{1111} & c_{1122} & c_{1133} & c_{1123} & c_{1131} & c_{1112} \\ & c_{2222} & c_{2233} & c_{2223} & c_{2231} & c_{2212} \\ & & c_{3333} & c_{3323} & c_{3331} & c_{3312} \\ & & & c_{2323} & c_{2331} & c_{2312} \\ & & & & c_{3131} & c_{3112} \\ \text{sym.} & & & & & c_{1212} \end{bmatrix} \tag{3.28}$$

For the isotropic Biot Material the unique components of this tensor can be written compactly as

$$c_{ijkl}^{SE} = \sum_{a,b=1}^{3} e_{ab}\Lambda_{ia}\Lambda_{ja}\Lambda_{lb}\Lambda_{kb} + h_{ab}\left(\Lambda_{ia}\Lambda_{jb}\Lambda_{ka}\Lambda_{lb} + \Lambda_{ia}\Lambda_{jb}\Lambda_{kb}\Lambda_{la}\right) \tag{3.29a}$$

where the sub-expressions are,

$$e_{ab} = \begin{cases} \dfrac{\gamma_2 - \gamma_1(\lambda_1 + \lambda_2 + \lambda_3 - 3 - \lambda_a)}{\lambda_a^3}, & a = b \\ \dfrac{\gamma_1}{\lambda_a\lambda_b}, & a \neq b \end{cases} \tag{3.29b}$$

$$h_{ab} = \begin{cases} 0, & a = b \\ \dfrac{\gamma_2 - \gamma_1(\lambda_1 + \lambda_2 + \lambda_3 - 3)}{\lambda_a\lambda_b(\lambda_a + \lambda_b)}, & a \neq b \end{cases}. \tag{3.29c}$$

These equations hold for both distinct and repeated stretches, so no special treatment is needed in different cases. The point-wise steps to implement the isotropic Biot material model in terms of the elasticity tensor is outlined in Algorithm 1. In a FEM code this procedure would be computed at every Gauss point during integration.

3.2.4 Linearization for implicit methods

The direct computation of $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$ is more involved than the previous use of the potential function. However, the direct computation outlined in the next section has

Algorithm 1 Computing the 2nd Elasticity Tensor for the Biot model

Precondition: Point-wise deformation information

1 Compute deformation gradient $\mathbf{F}$ ▷ (3.1)
2 $\mathbf{\Psi\Sigma\Lambda}^T \leftarrow \mathbf{F}$, by singular value decomposition ▷ (3.3)
3 Build $\mathbf{C}^{SE}$ ▷ (3.28) and (3.29)

the benefit of handling anisotropic material properties.

3.2.4.1 Anisotropic material

The computation of $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$ can be performed in many ways. This action is complicated by mixing tensors, Voigt notation, and finite element DOF. One naive possibility is to use symbolic computation (e.g. Mathematica) to compute expressions for each type of finite element. This results in amazingly large formulas and little generality. The approached offered here is to take the derivative of (3.7) with respect to a single DOF q_k, and employ the chain rule.

$$2\frac{\partial \boldsymbol{G}}{\partial q_k} = \frac{\partial \boldsymbol{U}}{\partial q_k}\boldsymbol{S} + \boldsymbol{U}\frac{\partial \boldsymbol{S}}{\partial q_k} + \frac{\partial \boldsymbol{S}}{\partial q_k}\boldsymbol{U} + \boldsymbol{S}\frac{\partial \boldsymbol{U}}{\partial q_k} \tag{3.30}$$

Using the chain rule, the derivative of the Biot stress can be expanded

$$\frac{\partial \boldsymbol{G}}{\partial q_k} = \frac{\partial \boldsymbol{G}}{\partial \boldsymbol{U}} : \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{F}} : \frac{\partial \boldsymbol{F}}{\partial q_k}. \tag{3.31}$$

The derivative of $\boldsymbol{F}$ is directly computed by finite element shape functions. If the displacement field is $\boldsymbol{u}(\boldsymbol{x}, t)$, and is approximated by the finite element interpolants $\mathbf{u} = \mathbf{Nq}$, then in Cartesian coordinates

$$\frac{\partial \boldsymbol{F}}{\partial q_k} \rightarrow \frac{\partial F_{ij}}{\partial q_k} = \frac{\partial}{\partial q_k}\left(\frac{\partial N_{il}}{\partial x_j} q_l\right) = \frac{\partial N_{il}}{\partial x_j}\left(\frac{\partial q_l}{\partial q_k}\right) = \frac{\partial N_{il}}{\partial x_j}\delta_{lk} = \frac{\partial N_{ik}}{\partial x_j}. \tag{3.32}$$

The computation of the derivative of the fourth order tensor $\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{F}}$ is involved. Several methods to accomplish this derivative have been proposed in the literature, such as Chen and Wheeler (1993); Rosati (1999) and Carroll (2004). The method used here is from the result of Chen and Wheeler (1993, Eqn. 10), which simplifies the construction by building the double contraction operator instead of the entire rank 4 tensor. The double contraction with a 2nd order tensor is

$$\boldsymbol{Y} := \mathrm{Tr}(\boldsymbol{U})\boldsymbol{I} - \boldsymbol{U} \tag{3.33a}$$

$$\boldsymbol{\Theta}_k := \frac{\partial \boldsymbol{U}}{\partial q_k} = \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{F}} : \frac{\partial \boldsymbol{F}}{\partial q_k} = \boldsymbol{R}^T \frac{\partial \boldsymbol{F}}{\partial q_k} - \frac{1}{\det(\boldsymbol{Y})} \boldsymbol{Y} \left(\boldsymbol{R}^T \frac{\partial \boldsymbol{F}}{\partial q_k} - \frac{\partial \boldsymbol{F}^T}{\partial q_k} \boldsymbol{R} \right) \boldsymbol{Y}\boldsymbol{U}. \tag{3.33b}$$

The final part of (3.31) yields

$$\begin{aligned}
\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{U}} &= \frac{\partial}{\partial \boldsymbol{U}} \left(\mathfrak{C} : (\boldsymbol{U} - \boldsymbol{I}) \right) \\
\frac{\partial G_{ij}}{\partial U_{kl}} &= \frac{\partial}{\partial U_{kl}} \left(C_{ijpq} U_{pq} - C_{ijpq} \delta_{pq} \right) = C_{ijpq} \frac{\partial U_{pq}}{\partial U_{kl}} \\
&= C_{ijpq} \frac{1}{2} \left(\delta_{pk} \delta_{ql} + \delta_{pl} \delta_{qk} \right) \\
&= \frac{1}{2} \left(C_{ijpq} \delta_{pk} \delta_{ql} + C_{ijpq} \delta_{pl} \delta_{qk} \right) = \frac{1}{2} \left(C_{ijkl} + C_{ijlk} \right) = C_{ijkl}.
\end{aligned}$$

The last step here is part of the usual major symmetry of the elasticity tensor.

$$\frac{\partial \boldsymbol{G}}{\partial \boldsymbol{U}} = \mathfrak{C} \tag{3.34}$$

Reducing the previous steps back into (3.30) and shifting some terms to the left of the equality provides

$$2\mathfrak{C} : \boldsymbol{\Theta}_k - \boldsymbol{\Theta}_k \boldsymbol{S} - \boldsymbol{S}\boldsymbol{\Theta}_k = \boldsymbol{U} \frac{\partial \boldsymbol{S}}{\partial q_k} + \frac{\partial \boldsymbol{S}}{\partial q_k} \boldsymbol{U}. \tag{3.35}$$

Here, all the terms on the left are known, and the only unknown is the $\frac{\partial \mathbf{S}}{\partial q_k} \in \mathbb{R}^{3\times 3}$. In linear algebra or control theory this type of equation is known as a linear Lyapunov

Algorithm 2 The computation of $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$, in Voigt notation, for an anisotropic Biot material with n_{ee} degrees of freedom.

Precondition: Point-wise deformation information

procedure COMPUTE_BIOT_DSDQ
  Compute deformation gradient $\mathbf{F}$ ▷ (3.1)
  $\mathbf{\Psi\Sigma\Lambda}^T \leftarrow \mathbf{F}$, by singular value decomposition ▷ (3.3)
  $\mathbf{U} \leftarrow \mathbf{\Lambda\Sigma\Lambda}^T$ and $\mathbf{R} \leftarrow \mathbf{\Psi\Lambda}^{\mathbf{T}T}$ ▷ (3.4)
  $\mathbf{G} \leftarrow \mathfrak{C} : (\mathbf{U} - \mathbf{I})$ ▷ (3.18)
  Solve for $\mathbf{S}$ ▷ (3.7)
  $\mathbf{Y} \leftarrow \mathrm{Tr}(\mathbf{U})\mathbf{I} - \mathbf{U}$ ▷ (3.33)
  for $k = 1$ to n_{ee} **do**
    $\frac{\partial\mathbf{F}}{\partial q_k}$ ▷ (3.32)
    $\mathbf{\Theta}_k$ ▷ (3.33)
    Solve for $\frac{\partial\mathbf{S}}{\partial q_k}$ ▷ (3.35)
    Gather $\frac{\partial\mathbf{S}}{\partial q_k}$ into column for Voigt numbering $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}(:, k)$
  end for
  return $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$
end procedure

equation. Jameson (1968) provides a direct solution for these linear problems in $\mathbb{R}^{3\times3}$. The steps to compute $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$, in Voigt notation, is shown in Algorithm 2.

3.2.4.2 Isotropic material

The special case of an isotropic material is computationally simpler than the anisotropic case. This is largely because all the strain and stress measures employed become coaxial. The relation between $\boldsymbol{G}$ and $\boldsymbol{S}$ simplifies to

$$\boldsymbol{S} = \boldsymbol{G}\boldsymbol{U}^{-1},$$

see (A.34) in §A.4.1 for the proof. This shortens (3.30) to

$$\frac{\partial \boldsymbol{S}}{\partial q_k} = \frac{\partial \boldsymbol{G}}{\partial q_k}\boldsymbol{U}^{-1} + \boldsymbol{G}\frac{\partial \boldsymbol{U}^{-1}}{\partial q_k}. \tag{3.36}$$

The $\frac{\partial \boldsymbol{G}}{\partial q_k}$ term will be computed in the same manner as before, expect with knowledge of (3.19) to make simplifications. The only new term is $\frac{\partial \boldsymbol{U}^{-1}}{\partial q_k}$, which can be expanded by the chain rule.

$$\frac{\partial \boldsymbol{U}^{-1}}{\partial q_k} = \frac{\partial \boldsymbol{U}^{-1}}{\partial \boldsymbol{U}} : \underbrace{\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{F}} : \frac{\partial \boldsymbol{F}}{\partial q_k}}_{\boldsymbol{\Theta}_k}$$

Employing the well-known identity (Holzapfel, 2000) gives

$$\frac{\partial \boldsymbol{U}^{-1}}{\partial q_k} = \frac{\partial \boldsymbol{U}^{-1}}{\partial \boldsymbol{U}} : \boldsymbol{\Theta}_k = -\boldsymbol{U}^{-1}\boldsymbol{\Theta}_k\boldsymbol{U}^{-1}. \tag{3.37}$$

Now substituting these relations into (3.36)

$$\begin{aligned}
\frac{\partial \boldsymbol{S}}{\partial q_k} &= \frac{\partial \boldsymbol{G}}{\partial q_k}\boldsymbol{U}^{-1} + \boldsymbol{G}\frac{\partial \boldsymbol{U}^{-1}}{\partial q_k} \\
&= \left(\boldsymbol{\mathfrak{C}} : \boldsymbol{\Theta}_k\right)\boldsymbol{U}^{-1} + \boldsymbol{G}\left(-\boldsymbol{U}^{-1}\boldsymbol{\Theta}_k\boldsymbol{U}^{-1}\right) \\
&= \left(\gamma_1 \mathrm{Tr}(\boldsymbol{\Theta}_k)\boldsymbol{I} + \gamma_2\boldsymbol{\Theta}_k\right)\boldsymbol{U}^{-1} - \underbrace{\boldsymbol{G}\boldsymbol{U}^{-1}}_{\boldsymbol{S}}\boldsymbol{\Theta}_k\boldsymbol{U}^{-1}.
\end{aligned}$$

Simplifying this give the explicit result as

$$\frac{\partial \boldsymbol{S}}{\partial q_k} = \left(\gamma_1 \mathrm{Tr}(\boldsymbol{\Theta}_k)\boldsymbol{I} + \gamma_2\boldsymbol{\Theta}_k - \boldsymbol{S}\boldsymbol{\Theta}_k\right)\boldsymbol{U}^{-1}. \tag{3.38}$$

This expression is much simpler to compute than solving the Lyapunov equation of (3.35). The overall process is also computationally simpler since there is no need to solve the $n_{ee} + 1$ Lyapunov equations for $\boldsymbol{S}$ or the components of its derivative. The ordered process is contained in Algorithm 3.

Algorithm 3 The computation of $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$, in Voigt notation, for an isotropic Biot material with n_{ee} degrees of freedom.

Precondition: Point-wise deformation information

procedure COMPUTE_BIOT_DSDQ_ISO
  Compute deformation gradient $\mathbf{F}$ ▷ (3.1)
  $\mathbf{\Psi\Sigma\Lambda}^T \leftarrow \mathbf{F}$, by singular value decomposition ▷ (3.3)
  $\mathbf{U} \leftarrow \mathbf{\Lambda\Sigma\Lambda}^T$ and $\mathbf{R} \leftarrow \mathbf{\Psi\Lambda}^{\mathbf{T}T}$ ▷ (3.4)
  $\mathbf{G} \leftarrow \gamma_1 \mathrm{Tr}(\mathbf{U}-\mathbf{I})\mathbf{I} + \gamma_2(\mathbf{U}-\mathbf{I})$ ▷ (3.19)
  $\mathbf{S} \leftarrow \mathbf{U}^{-1}\mathbf{G}$ ▷ (A.34)
  $\mathbf{Y} \leftarrow \mathrm{Tr}(\mathbf{U})\mathbf{I} - \mathbf{U}$ ▷ (3.33)
  for $k = 1$ to n_{ee} **do**
    $\frac{\partial\mathbf{F}}{\partial q_k}$ ▷ (3.32)
    $\mathbf{\Theta}_k$ ▷ (3.33)
    $\frac{\partial\mathbf{S}}{\partial q_k}$ ▷ (3.38)
    Gather $\frac{\partial\mathbf{S}}{\partial q_k}$ into column for Voigt numbering $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}(:,k)$
  end for
  return $\frac{\partial\{\mathbf{S}\}}{\partial\mathbf{q}}$
end procedure

3.3 Implementation using geometrically exact finite elements

3.3.1 Overview

The goal of the tools constructed here are to explore fluid-structure interaction problems, like flapping wings. Solid finite elements are used since the supporting theory and technology is widely known. They provide a foundation to build a framework that many different types of structures, wings, material models, etc. can be tested. The implementation is general enough to handle structural elements as well, and their integration is a possible avenue for future work. Solid models were chosen since they provide finite thickness, fundamentally have the fewest assumptions, and they can be widely adapted for a variety of continuum-mechanics based response models. The relative cost of using solid elements in a large-scale CFD simulation is very small.

The implementation technology employed here is largely based on Hughes (2000) for the assembly and shape functions and Belytschko, Liu, and Moran (2000) for dealing with nonlinear models. Mesh generation is designed around the open-source software Gmsh (Geuzaine and Remacle, 2009). The elements implemented are isoparametric quadratic hexahedra for the volume of the body. These 27-node displacement based elements were selected since they will not suffer from locking like linear elements. Surface elements used for the FSI and other loading are 9 node quadrilaterals. Each quadrilateral is coincident to a single face of a corresponding hexahedron. See A.3.1 for the technical details of the elements such as nodal numbering and shape functions.

The implementation contains both Kirchhoff and Biot material models for isotropic properties. The implementation of the Biot material was selected to be Algorithm 3. This method was selected since it provides a pattern for future code development for problems with anisotropic materials.

3.3.2 Equations of motion

The description of motion implemented is a weak-form of momentum conservation often called the Total Lagrangian formulation. It is a Lagrangian method where everything is expressed in terms of the reference configuration. In the usual finite element way, the virtual work of the body is expressed as

$$\delta W_{\text{int}} - \delta W_{\text{ext}} = 0 \tag{3.39}$$

The virtual internal work can be expressed in terms of any work-conjugate pair. The simplest pair to work with is $(\boldsymbol{E}, \boldsymbol{S})$, which in Voigt notation is

$$\delta W_{\text{int}} = \int_{\Omega_0} \{\delta \mathbf{E}\}^T \{\mathbf{S}\} \, dV_0 = \sum_{e=1}^{n_{el}} \delta W_{\text{int}}^e. \tag{3.40}$$

The external work can be viewed as the sum of work due to body forces, such as acceleration and gravity, and surface tractions.

$$\delta W_{\text{ext}} = \delta W_{\ddot{\boldsymbol{u}}} + \delta W_{\boldsymbol{f}_{\text{ext}}} \tag{3.41}$$

The surface forces will be treated in §4.2.3. The acceleration term can be written as

$$\delta W_{\ddot{\boldsymbol{u}}} = \int_{\Omega_0} \delta \boldsymbol{u} \cdot (-\ddot{\boldsymbol{u}} \rho_0) \; dV_0. \tag{3.42}$$

Using the usual finite element shape function approximations from (3.14)

$$\boldsymbol{u} \to \mathbf{u} = \mathbf{N}\mathbf{q}, \text{ therefore } \ddot{\boldsymbol{u}} \to \mathbf{u} = \mathbf{N}\ddot{\mathbf{q}} \tag{3.43}$$

gives the definition for the consistent mass matrix.

$$\delta W_{\ddot{\boldsymbol{u}}} = \int_{\Omega_0} \delta \mathbf{q}^T \mathbf{N}^T (-\mathbf{N}\ddot{\mathbf{q}} \rho_0) \; dV_0 = -\delta \mathbf{q}^T \int_{\Omega_0} \rho_0 \mathbf{N}^T \mathbf{N} \; dV_0 \, \ddot{\mathbf{q}} \tag{3.44}$$

$$\mathbf{M} = \int_{\Omega_0} \rho_0 \mathbf{N}^T \mathbf{N} \; dV_0 \tag{3.45}$$

Notable features of this matrix are that it is symmetric, and constant. The symmetry allows for computational efficiency in storage and inversion. The fact that it is constant means it only needs to be built once. In explicit dynamic algorithms it needs to only be decomposed once as well. This formulation is still consistent with large motions, and makes no implied assumptions about the body. The principle of the conservation of mass can be used to show that this constant mass matrix in the Total Lagrangian formulation is equivalent to the deformation dependent mass matrix by other formulations like the updated Lagrangian form (Belytschko *et al.*, 2000).

Combing the definitions of (3.10) and (3.44) into (3.39) gives

$$0 = \delta\mathbf{q}^T \left(\mathbf{f}_{\text{int}} + \mathbf{M}\ddot{\mathbf{q}} - \mathbf{f}_{\text{ext}}\right) .$$

This can be recast as the semi-discrete equation of motion, with the inclusion of assuming linear damping.

$$\mathbf{M}\ddot{\mathbf{q}}(t) + \mathbf{D}\dot{\mathbf{q}}(t) + \mathbf{f}_{\text{int}}(\mathbf{q}, t) = \mathbf{f}_{\text{ext}}(\mathbf{q}, t) \tag{3.46}$$

The calculation of $\mathbf{f}_{\text{int}}$ is determined by how the selected material model computes $\boldsymbol{S}$. Essential boundary conditions have not yet been applied to this equation and are needed, along with initial conditions, to fully pose the problem.

3.3.3 Application of essential boundary conditions

The degrees of freedom of the entire body are ordered during the preprocessing of the mesh to place the restrained components at the end of the global list. Thus if $\mathbf{q}$ are the DOF for the entire body, then this list is partitioned as

$$\mathbf{q} = \left[\begin{array}{c} \bar{\mathbf{q}} \\ \hline \mathbf{v} \end{array}\right] \tag{3.47}$$

where $\bar{\mathbf{q}}$ are the unrestrained DOF, and $\mathbf{v}$ are the DOF that have some essential boundary condition applied to them. Here, $\mathbf{v}(t)$ will be fully defined $\mathcal{C}^2$-functions of time that prescribe the motion of points on the body. This permits the direct partitioning of the mass and damping matrices in (3.46).

$$\left[\begin{array}{cc} \bar{\mathbf{M}} & \mathbf{M}_v \\ \mathbf{M}_v^T & \mathbf{M}_{vv} \end{array}\right] \left[\begin{array}{c} \ddot{\bar{\mathbf{q}}} \\ \hline \ddot{\mathbf{v}} \end{array}\right] + \left[\begin{array}{cc} \bar{\mathbf{D}} & \mathbf{D}_v \\ \mathbf{D}_v^T & \mathbf{D}_{vv} \end{array}\right] \left[\begin{array}{c} \dot{\bar{\mathbf{q}}} \\ \hline \dot{\mathbf{v}} \end{array}\right] + \left[\begin{array}{c} \bar{\mathbf{f}}_{\text{int}}(\mathbf{q}, t) \\ \hline \mathbf{f}^v_{\text{int}}(\mathbf{q}, t) \end{array}\right] = \left[\begin{array}{c} \bar{\mathbf{f}}_{\text{ext}}(\mathbf{q}, t) \\ \hline \mathbf{f}^v_{\text{ext}}(\mathbf{q}, t) \end{array}\right] \tag{3.48}$$

Extracting only the top equation for the unconstrained DOF gives the equation of motion with boundary conditions applied

$$\bar{\mathbf{M}}\ddot{\bar{\mathbf{q}}}(t) + \bar{\mathbf{D}}\dot{\bar{\mathbf{q}}}(t) + \bar{\mathbf{f}}_{\text{int}}(\mathbf{q}, t) = \mathbf{h}(\mathbf{q}, t) \tag{3.49a}$$

$$\mathbf{h}(\mathbf{q}, t) := \bar{\mathbf{f}}_{\text{ext}}(\mathbf{q}, t) - \mathbf{M}_v\ddot{\mathbf{v}}(t) - \mathbf{D}_v\dot{\mathbf{v}}(t). \tag{3.49b}$$

Here the over bars should emphasize that these DOF are the free DOF, and the internal and external forces do depend on all the DOF. During the linearization of $\mathbf{f}_{\text{int}}$ no additional matrix-partition terms need to be included in $\mathbf{h}$ since their contributions are already present in $\bar{\mathbf{f}}_{\text{int}}(\mathbf{q}, t)$.

3.3.4 Efficient sparse assembly

The mass and tangent stiffness matrices for unstructured meshes are unstructured sparse matrices. The implementation currently does not permit adaptation of the body's mesh, so the connectivity and assemble procedures will not change during a simulation. Therefore a compact and efficient method should be used to assemble $\mathbf{M}$, $\mathbf{K}$, and optionally the damping $\mathbf{D}$. There are several common storage schemes for sparse matrices, and each has its benefits (Saad, 2003). Different storage schemes make particular matrix operations more efficient. For example, the compressed sparse row (CSR) format stores the entries of the matrix in a rank 1 array sorted by rows with two additional rank 1 arrays as indices. This enables very efficient sparse-matrix dense-vector products. Due to this, CSR is commonly employed in iterative solvers such as Kyrlov methods. For the types of problems of interest here, the relative computational cost of the solid body compared to the fluid is small, so direct solution methods were sought. Although known to not scale as well as iterative methods, direct methods have the benefit of being far more robust if they are computationally available. The solver used in the current implementation is

from the open-source library SuperLU (Li *et al.*, 1999; Li, 2005) based on a novel approach to partial pivoting in Gaussian elimination (Demmel *et al.*, 1999). The serial version of this library is sufficient for moderate mesh sizes and is comparable to the UMFPACK solver (Davis, 2004) used in MATLAB. SuperLU and UMFPACK are both written in the C language, but SuperLU has a simpler interface for Fortran. The serial version of SuperLU requires the compressed sparse column storage (CSC) format, which is a common requirement of most direct methods.

Leveraging the fixed nature of the mesh, there are several computations that can be preprocessed to aid in speeding up the simulation. A custom MATLAB script reads in the text file of Gmsh node and element information and outputs a hdf5 file (HDF Group, 2013) to be read into the main Fortran codes. These operations are expensive but the computational cost is not important since it is only performed once per mesh. The first major operation is to reorder the numbering of the unrestrained DOF for solution efficiency. Reordering is known to affect the fill-in of the LU decomposition. Less fill-in means less storage needed during factorization, which implies a faster solve from SuperLU. In MATLAB this reordering is performed by first assembling the entire tangent stiffness matrix with Boolean operators. This gives initial sparsity pattern, the unrestrained partition of this matrix is extracted, and then reorder using a minimum degree method such as `amd`. The integer map of the connectivity to the equation number, classically called the location matrix, is then rebuilt using the reordering information.

The location matrix is classically a mapping from local equation number p of element e to global equation number P, e.g. $LM(p, e) \mapsto P$. During the assembly process after a mass/stiff matrix for element e has been generated it is superimposed into the global matrix by mapping each component p_1 and p_2 through LM. In the case of sparse formatting this is an issue. In either CSR or CSC format the (P_1, P_2) component of the matrix needs to be mapped to a linear array, index M,

for storage. The naive approach to generate a mapping from $(P_1, P_2) \mapsto M$ but this would require the storage of a dense set of integers $\mathbb{N}^{n_{dofs} \times n_{dofs}}$, the same dimensions as the sparse global matrix. This would probably be far too large to store for even moderately sized meshes. The method employed in the Fortran codes here setup a slightly simpler mapping. Instead of requiring the processing of LM twice to compute (P_1, P_2), a single rank-3 array is used. This additional integer array, called LM_c, is used to map each individual component from each element matrix to the linear index of the global matrix.

$$LM_c(p_1, p_2, e) \mapsto M \tag{3.50}$$

The size of $LM_c \in \mathbb{N}^{n_{ee} \times n_{ee} \times n_{el}}$ is far smaller than $\mathbb{N}^{n_{dofs} \times n_{dofs}}$. Using LM_c permits the direct construction of a matrix in CSR or CSC formats. This bypasses the steps of constructing a sparse matrix in a redundant coordinate format, and then compressing it to the desired format for a particular solver every time the matrix is assembled.

3.3.5 Numerical integration of the Biot material

The theoretical construction of the Biot material model uses the polar decomposition to acquire the stretch tensor $\boldsymbol{U}$. Couple these computations to the implementation of isoparametric finite elements, and the integrands involved are not polynomials. This poses a problem for numerical integration of the internal forces (3.12) and the stiffness matrix (3.13). By contrast, the stress field of the Kirchhoff material is a polynomial,[2] and is exactly integrated using standard Gauss-Legendre quadrature.

[2]For isoparametric elements the Kirchhoff material is a rational function in general. Since the element used here is a cube, the isoparametric transformation only scales the coordinates and is a polynomial.

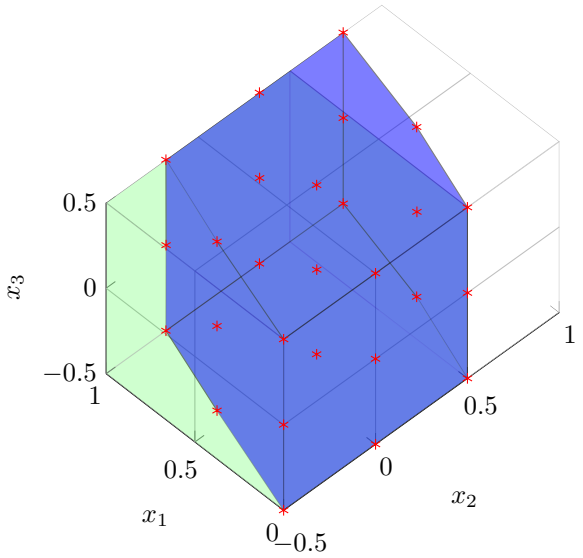


Figure 3.5: Demonstration of a single finite element to determine integration order. The face at $x_1 = 0$ is fixed, while the displacements on the opposite face are constrained to $(u_1, u_2, u_3) = (0, u_2, 0)$ and a load is applied in the positive x_2 direction.

Deformations with curvature increase the computational influence of $\boldsymbol{R}$ and are subsequently farther from polynomials. Fixing one face of a single finite element while tangentially loading the opposite face provides a test case to explore these issues. In finite elasticity this is not simple shear, i.e. it does not produce a homogeneous deformation. Figure 3.5 contains a representative body made up of one finite element that is fixed on one face and loaded tangentially on the opposite face. This closely resembles simple shear, but provides finite deformation with curvature. The numerical integration techniques implemented in the code for the natural cube are based on the tensor products of Gauss-Legendre quadrature common in one-dimension. Stroud (1971) is a classical text and outlines that these points are not computationally optimal for the cube. However, they do provide good accuracy and a constructive method to build arbitrarily high-ordered integrators. Other methods, if desired, can easily be implemented to extend the code in the future.

It is well known that Gaussian quadrature is exact for polynomials up to an

appropriate degree. In the case of the Kirchhoff material implemented in a quadratic finite element, the equations should be exactly integrated using 5 quadrature points in each direction. This gives the total number of points where the stress needs to be computed inside each element as $5^3 = 125$. To examine how the Biot model compares in this example case is conducted over a range of integration points. Figure 3.6 contains a graph of the relative error

$$\text{error} = \left| \frac{q_{\text{load}} - q_{\text{ref}}}{q_{\text{ref}}} \right| \tag{3.51}$$

where q_{load} is the average nodal displacement of the loaded face in the x_2 direction. The reference displacement q_{ref} is computed by using 10 Gauss points in each direction. The load is incremented to the same value for all the models. The error of the Kirchhoff material drop to roughly machine precision right at 5 points as expected. The Biot material however, takes significantly more integration points to achieve convergence. This example is for a very large displacement and in practice it was found that for most common problems it was unnecessary to increase the number of integration points past 5. Selective under-integration is a common tool used in finite element methods to prevent locking since it makes the body appear softer. Keeping the integration to only 5 points in each direction effectively does the same thing.

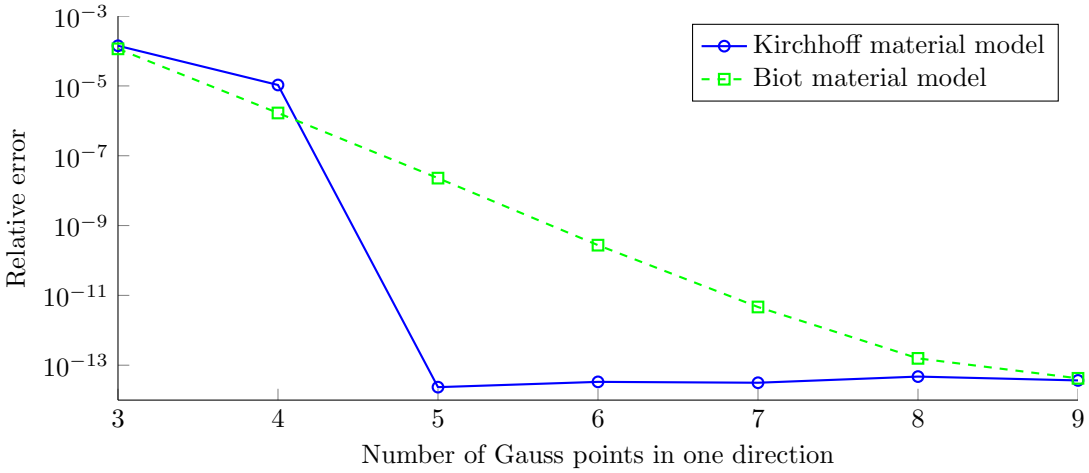


Figure 3.6: Relative convergence of a single finite element in simple shear as the number of integration points is increased.

Chapter 4

Fluid-structure interactions

4.1 Overview

Fluid-structure interactions are a widely studied field with an immense amount of techniques tailored to specific problem domains. A partitioned method is a particular way to handle mixed formulation problems. Here, the fluid equations are discretized with a fixed Eulerian grid, and the body is discretized on a Lagrangian mesh. A recent survey of this type of method was compiled by Degroote (2013). Several major variations of the method are presented along with a multitude of references. The partitioned scheme employed here is in essence the same fixed-point iteration that was used in Preidikman (1998), Yang *et al.* (2008), and Vanella *et al.* (2009) but tailored for finite elements representing the body.

The definition of Felippa, Park, and Farhat (2001) nicely summarizes these types of methods.

> *Partitioned treatment.* The field models are computationally treated as isolated entities that are separately stepped in time. Interaction effects are viewed as forcing effects that are communicated between the individual components using prediction, substitution, and synchronization techniques.

Furthermore, they state that no technical argument can be made for the superiority of partitioned methods verses *monolithic methods* (when all the equations are solved simultaneously). Instead, each method is better suited to certain types of problems and implementations.

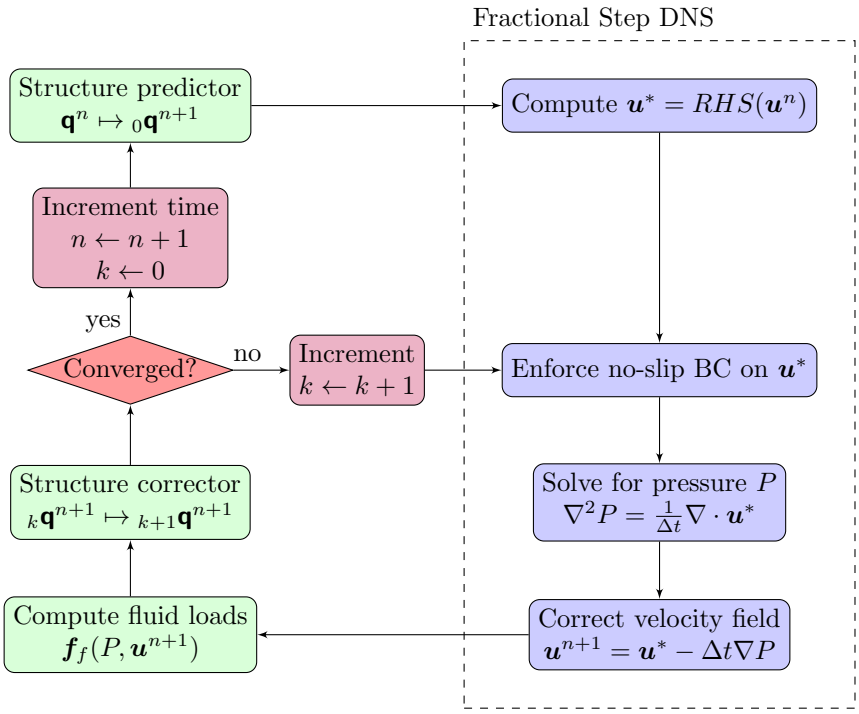


Figure 4.1: Procedure diagram for the partitioned FSI algorithm.

There are many varieties of partitioned methods based on a prediction – correction model. The methods differ in what is used for the prediction, and if the correction is used repeatedly or staggered in time. The method implemented here, as outline in Fig. 4.1, is sub-iterated until convergence of the entire system's equilibrium is achieved. The fluid solver is based on same explicit fractional step method used through the previous chapters, but implemented inside the FLASH framework (ASC Flash Center, 2012; Daley, Vanella, Dubey, Weide, and Balaras, 2012). The large scale high performance computing (HPC) framework is designed to tackle extremely large problem domains on finite-difference grids. FLASH can use either uniform gridding techniques or adaptive mesh refinement (AMR) based on the PARAMESH library (MacNeice, Olson, Mobarry, de Fainchtein, and Packer, 2000).

Each time step begins by predicting the states of the structure, and computing the position, velocity, and acceleration fields of the body's wet surface. Then the fluid velocity field $\boldsymbol{u}^*$ is computed. In the usual manner of fractional step methods,

this field is not divergence free. The surface kinematics of the body are then applied to the fluid grid points around the body in a Lagrangian fashion as implemented by Vanella (2010). Calculating the pressure P presents the most expensive step in the calculation. This elliptical problem, often referred to as the *Poisson problem*, is discretized to become a set a simultaneous linear equations. The efficient calculation of the pressure represents one of the largest hurtles to large scale solutions (Daley *et al.*, 2012). Once the pressure gradient is computed, the corrected (or end-of-step) velocity $\boldsymbol{u}^{(n+1)}$ is calculated and stored. The velocity and pressure information are then used to compute the surface forces on the body. A corrector procedure then computes an updated estimate of the states of the body. If the states have not changed within some tolerance, then the velocity field of the immersed boundary conditions are recomputed and the cycle repeats. Once the convergence criterion has been satisfied, time is incremented and the outer loop begins again with a prediction of the structure using the previous fluid load.

The implementation issues of the predictor-corrector method used here are mostly surrounding the treatment of the body, since the coding for the fluid model is already in place. This entails both the time integration of the body as well as the construction of the forcing terms as boundary conditions on each partitioned field. Inside the FLASH architecture there is an entire unit of the code dedicated to Lagrangian particle tracking known as `PARTICLES`. The previous uses of these particles range from physics simulations, to convecting massless particles for event tracing. For the FSI implementation, they will serve as the method of communication between the fluid domain and the structural domain.

The first issue to address is the imposition of boundary conditions since this is general to any body, and any time marching scheme used for that body. Section 4.2 begins with the use of `PARTICLES` and their relation to patches of the body's surface area. The implementation of imposing the kinematic field of the body on the fluid is

outlined in §4.2.2. Building the surface tractions on the body, and projecting them onto the DOF of the structure are shown in §4.2.3. Finally, in §4.3 the construction of two integrators specifically designed for FSI are formulated.

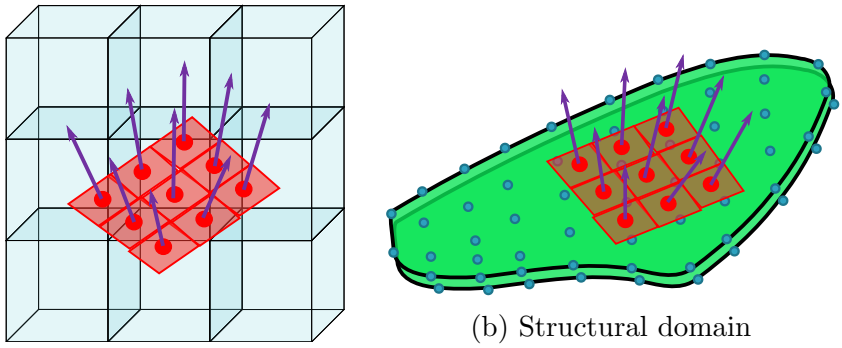
4.2 Imposition of boundary conditions

4.2.1 Particles and patches

Inside the FLASH code, the `PARTICLES` unit is a well apportioned framework for working with Lagrangian points distributed across the Eulerian domain. The distribution of the particles on the HPC cluster is performed by FLASH. The immersed boundary unit called `ImBound` uses the information of each particle to enforce the no-slip condition. The use of `PARTICLES` then is to cover the body's surface with particles whose kinematics are prescribed by the surface of the body. These particle points are used for both parts of the communication of boundary conditions: forcing the fluid, and forcing the body.

Figure 4.2a shows a small collection of particles inside the fixed fluid grid. These particles each represent a small patch of the surface area on the surface of a body, Fig. 4.2b. It is the information of the particles that connects the fluid and structural domains. Each patch of surface must be near the same size as the fluid grid spacing. Therefore the spacing of particles is determined by the fluid grid since for most problems the fluid mesh will be much finer than the body's mesh. This permits the meshing of the structure to be independent of the fluid grid.

A designer of the body would need only to be concerned with sufficient spatial resolution for the body's deformation. This is in contrast to previous immersed boundary implementations. In Vanella (2010), a rigid wing with the planform of a *Musca domesica* was defined using 381 662 triangles. If each triangle was mapped to the face of a tetrahedron finite element, the number of DOF in a relatively simple



(a) Fluid domain

(b) Structural domain

Figure 4.2: A representative region of fluid particles in the fluid domain, and these same particles in the structural domain. (a) The fluid domain particles are used to represent the kinematics of the body's surface. (b) Likewise, the particles are interpreted by the body as patches of constant applied traction.

body would be staggering. The method implemented here using particles avoids this complication by grouping particles by surface element. Now a subset of the particles is indexed to a surface element, and this mapping is structured to allow for memory efficiency and calculation speed.

The mapping of each particle to an individual patch of surface on a the body is shown in Fig. 4.3. Here $\hat{\boldsymbol{n}}$ is the outward facing unit vector at center of the patch, and $\{\hat{\boldsymbol{t}}_1, \hat{\boldsymbol{t}}_2\}$ are a pair of vectors tangent to the surface. The surface normal is computed by the cross-product of two independent vectors on the surface of the body. To ensure that this calculation results in outward facing normals a check is performed during mesh pre-processing, see §A.3.4 for details. The distributed force $\boldsymbol{f}_f$ is defined in the global frame. The area of each surface patch is A_p. The calculations to determine the kinematics of the deformed surface are performed at the center point of the patch. Since the spacing of particles is the same as the fluid grid, then it is assumed that $\boldsymbol{f}_f$ is constant on each patch.

The local ordering of the patches on each element is constructed by using Fig. 4.4. First the length of the deformed element is calculated, and compared with the

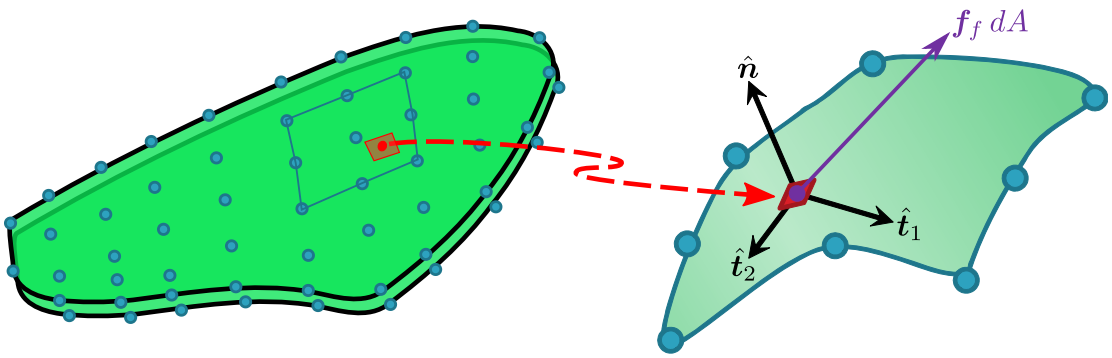


Figure 4.3: Discrete surface element of the body. The triad $\{\hat{\boldsymbol{n}}, \hat{\boldsymbol{t}}_2, \hat{\boldsymbol{t}}_2\}$ is a local set of unit vectors on the patch p, with differential area dA and surface traction $\boldsymbol{f}_f$.

local fluid grid to determine the spacing of particles on the surface element. There are n_ξ and n_η particles in the ξ-direction and η-direction, respectively. Each particle is assigned a local index p, and this integer uniquely places the patch in the 2D grid (k, j) where $k \in [1, n_\xi] \subset \mathbb{N}$ and $j \in [1, n_\eta] \subset \mathbb{N}$. Defining the mapping

$$p = (j-1)n_\xi + k \tag{4.1}$$

which provides the inverse

$$j = \text{ceiling}\left(\frac{p}{n_\xi}\right) \tag{4.2a}$$

$$k = p - (j-1)n_\xi. \tag{4.2b}$$

These definitions can be used to construct the bounds of the patch p

$$\xi \in [\xi_1, \xi_2]: \qquad \xi_1 = (k-1)\frac{2}{n_\xi} - 1, \qquad \xi_2 = (k-0)\frac{2}{n_\xi} - 1 \tag{4.3a}$$

$$\eta \in [\eta_1, \eta_2]: \qquad \eta_1 = (j-1)\frac{2}{n_\eta} - 1, \qquad \eta_2 = (j-0)\frac{2}{n_\eta} - 1 \tag{4.3b}$$

as well as the center point of the patch, where the particle position, velocity, and

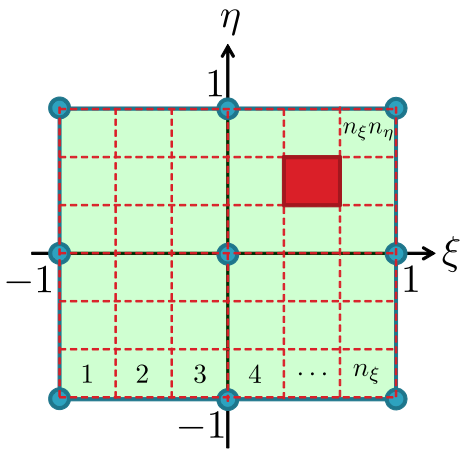


Figure 4.4: Natural domain of the surface element, showing a single particle patch.

acceleration are calculated.

$$\xi_p = \frac{\xi_2 + \xi_1}{2} \tag{4.4a}$$

$$\eta_p = \frac{\eta_2 + \eta_1}{2} \tag{4.4b}$$

4.2.2 Application of no-slip condition to the fluid

The `ImBound` unit already contains the moving-least-squares interpolations to force the fluid grid (Vanella, 2010). The FLASH code was setup to expect that the particles data-structure is populated with the position, velocity, and acceleration of each particle. This is readily accomplished in the FEM solid since all of these quantities can be computed from the DOF. Each particle knows the natural coordinates (ξ_p, η_p) of the surface element it belongs to. Therefore the calculation of the surface kinematics for particle p on surface element e begins with extracting the node locations and DOF for the element and storing them in $\mathbf{x}^e$, $\mathbf{q}^e$ respectively. The time marching schemes also store the time derivatives of the DOF. Using the

standard finite element shape functions, these quantities are directly computed by

$$\mathbf{r}_p = \mathbf{N}(\xi_p,\, \eta_p)\,(\mathbf{x}^e + \mathbf{q}^e) \tag{4.5a}$$

$$\dot{\mathbf{r}}_p = \mathbf{N}(\xi_p,\, \eta_p)\dot{\mathbf{q}}^e \tag{4.5b}$$

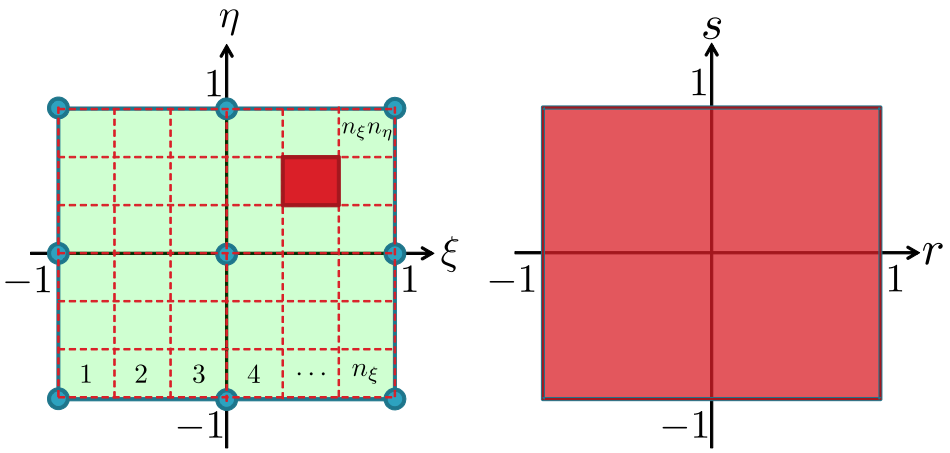
$$\ddot{\mathbf{r}}_p = \mathbf{N}(\xi_p,\, \eta_p)\ddot{\mathbf{q}}^e. \tag{4.5c}$$

Where $\mathbf{r}_p$ is the position of the particle in the global frame, and the over dots are its derivatives with respect to time.

4.2.3 Application of surface stress to the body

The consistent projection of the fluid stress onto the DOF of the body is somewhat involved. In order for the `PARTICLES` unit to construct the constant distributed force $\boldsymbol{f}_f$ on the surface of the patch the normal vector $\hat{\boldsymbol{n}}$ and current area A_P need to be built. Recall that the surface elements of the body are quadratic, and could be highly deformed. The problem is then to determine how to integrate over a small patch of the element consistently to project the loads onto the DOF. Numerical integration of arbitrary orders on the square can be accomplished with the usual tensor product rules, so a linear transformation is used to change the variables. These rules are not optimal, but they are simple to compute and implement on the natural square. Figure 4.5a shows the small patch in the natural coordinates $(\xi,\, \eta)$, while Fig. 4.5b displays the same patch transformed into coordinates $(r,\, s)$. This permits general quadrature-rules based on the $(-1, -1)$ interval to be used.

The kinematics of the deformed surface element are computed using the usual FEM shape functions. The transformation of $(\xi,\, \eta) \to (r,\, s)$ can be seen as a bilinear interpolation. If the bounding box of the patch is $\xi \in [\xi_1, \xi_2]$ and $\eta \in [\eta_1, \eta_2]$, then



(a) Natural domain of the surface element, showing a single particle patch.

(b) Natural domain of the particle patch.

Figure 4.5: Computational domains for the surface element, where $(\xi,\, \eta)$ are the natural coordinates of the isoparametric finite element and $(r,\, s)$ are the natural coordinates of the red patch p.

using the point-slope formula of a line

$$\xi - \xi_1 = \frac{\xi_2 - \xi_1}{1 - (-1)}(r - (-1))$$

provides a form of $\xi(r)$ and $\eta(s)$ as

$$\xi(r) = \left(\frac{\xi_2 - \xi_1}{2}\right) r + \frac{\xi_2 + \xi_1}{2} \tag{4.6a}$$

$$\eta(s) = \left(\frac{\eta_2 - \eta_1}{2}\right) s + \frac{\eta_2 + \eta_1}{2}. \tag{4.6b}$$

A differential area on the $(\xi,\, \eta)$ square would be

$$d\xi\, d\eta = \frac{1}{4}\left(\xi_2 - \xi_1\right)\left(\eta_2 - \eta_1\right)\ dr\, ds\,. \tag{4.7}$$

Let a point on the surface element be defined by

$$\mathbf{r} = \mathbf{N}(\xi,\, \eta)\,(\mathbf{x}^e + \mathbf{q}^e) \tag{4.8}$$

in the same manner as before. Then the actual area of the deformed patch is computed by

$$A_p = \int_{\partial\Omega_p} dA = \int_{\eta_1}^{\eta_2}\int_{\xi_1}^{\xi_2} |\mathbf{r}_{,\xi} \times \mathbf{r}_{,\eta}|_2 \; d\xi\, d\eta \tag{4.9}$$

$$= \frac{1}{4}\,(\xi_2 - \xi_1)\,(\eta_2 - \eta_1)\int_{-1}^{1}\int_{-1}^{1} |\mathbf{r}_{,\xi}(\xi(r),\, \eta(s)) \times \mathbf{r}_{,\eta}(\xi(r),\, \eta(s))|_2 \; dr\, ds\,. \tag{4.10}$$

The term $|\mathbf{r}_{,\xi} \times \mathbf{r}_{,\eta}|_2$ is often referred to as the surface Jacobian in the change of variables theorem. The unit normal vector $\hat{\boldsymbol{n}}$ is constructed directly as

$$\mathbf{n} = \left.\frac{\mathbf{r}_{,\xi} \times \mathbf{r}_{,\eta}}{|\mathbf{r}_{,\xi} \times \mathbf{r}_{,\eta}|_2}\right|_{(\xi=\xi_p,\, \eta=\eta_p)}. \tag{4.11}$$

The area and normal vectors are needed by the `ImBound` routines to compute the stress $\boldsymbol{f}_f$. Once the value of $\boldsymbol{f}_f$ is updated, it can be projected onto the DOF of the surface element. Since the domains of each patch have compact support the virtual work of the traction can be decomposed as

$$\delta W_f = \int_{\partial\Omega} \delta\boldsymbol{r} \cdot \boldsymbol{f}_f \; dA = \delta\mathbf{q}^T \sum_{p=1}^{n_\xi\, n_\eta} \int_{\partial\Omega_p} \mathbf{N}^T \mathbf{f}_f \; dA. = \delta\mathbf{q}^T \sum_{p=1}^{n_\xi\, n_\eta} \mathbf{f}_p$$

Therefore the forces of patch p, labeled as $\mathbf{f}_p$, are superimposed with the contributions from all the other patches on the surface element. This computation of the force can be performed in parallel with a summed reduction. Looking at a single patch, the

force is computed using some of the relations used earlier to compute the area.

$$\mathbf{f}_p = \int_{\partial\Omega_p} \mathbf{N}^T \mathbf{f}_f \; dA \tag{4.12}$$

Using the same change of variables as before, the integration is recast into the $(r,\, s)$ domain as

$$\mathbf{f}_p = \frac{1}{4}\left(\xi_2 - \xi_1\right)\left(\eta_2 - \eta_1\right) \int_{-1}^{1}\int_{-1}^{1} \mathbf{N}(\xi(r),\, \eta(s))^T \left|\mathbf{r}_{,\xi}(\xi(r),\, \eta(s)) \times \mathbf{r}_{,\eta}(\xi(r),\, \eta(s))\right|_2 \; dr\, ds\, \mathbf{f}_f. \tag{4.13}$$

The calculation is carried out by particle p and then globally reduced into the generalized loading on the body.

The force due to the pressure term(Vanella, 2010) was linearized such that it produced a non-symmetric contribution to the tangent stiffness matrix. However it was not found to change the number of iterations significantly while adding an expensive calculation. It remains in the code for future trials, but it is not in current usage.

4.3 Time integrators for the body

4.3.1 Adams predictor–corrector method

The Adams family of methods is a wellknown subset of linear multistep methods. The methods are comprised of the weighted superposition of the states and their derivatives in time. Methods that only use information from previous times are called explicit methods, and name associated to these is commonly Adams-Bashfourth methods. Implicit methods, or Adams-Moulton methods, use both previous as well as the current time information making them more expensive. Each type of method

has particular strengths: explicit methods are computationally simpler and cheaper, while implicit methods usually have larger regions of stability for a given order.

It is common across many numerical problems to couple these explicit and implicit methods in a predictor-corrector fashion. There are several possible variations based on the Predict (P) – Evaluate (E) – Correct (C) – Evaluate (E) model. During a prediction (P), old values of the states are used to estimate the states at the new time. Evaluate (E) means to construct the action of the vector field at this new value of the states. The corrector (C) uses an implicit method with the approximate value of the new states to arrive at an improved estimate for the new states. The method employed in this work is to use the so-called PE(CE)$^\infty$ model, where the corrections are performed repeatedly until some measure of error is achieved.

In the field of computational fluid dynamics, the use of Adams methods dates back to the first widely employed fractional step method (Kim and Moin, 1985). The use of multistep methods for FSI has been demonstrated for low order fluids models (Preidikman, 1998; Preidikman and Mook, 1998) as well as viscous Navier-Stokes problems (Yang, Preidikman, and Balaras, 2008).

4.3.1.1 Formulation

All the theory for multistep methods is built on first order equations, so the first order of business is to transform the FEM equation of motion to state-space. Dropping the over-bars for simplicity, (3.49) is restated as

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{f}_{\text{int}}(\mathbf{q}) = \mathbf{h}(\mathbf{q}, t) \tag{4.14}$$

where $\mathbf{h}(\mathbf{q}, t)$ is the external forcing, including contributions from boundary conditions.

$$\mathbf{h}(\mathbf{q}, t) = \mathbf{f}_{\text{ext}}(\mathbf{q}, t) - \mathbf{M}_v \ddot{\mathbf{v}}(t) - \mathbf{D}_v \dot{\mathbf{v}}(t) \tag{4.15}$$

Now let the states of the system be defined as the partitioned column

$$\mathbf{y} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}. \tag{4.16}$$

Defining the first-order evolution equation as

$$\dot{\mathbf{y}} = \mathbf{g}(\mathbf{y}, t) = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{M}^{-1}\left(\mathbf{h}(\mathbf{q}, t) - \mathbf{D}\dot{\mathbf{q}} - \mathbf{f}_{\text{int}}(\mathbf{q})\right) \end{bmatrix}. \tag{4.17}$$

This equation is nice to compute with since $\mathbf{M}$ is a constant, and its inverse need only be computed once and stored. The remaining terms on the left-hand side are assembled in the usual FEM manner. The coefficients for variable size timesteps are derived and computed for several order in §A.2.3. The coefficients are messy and not repeated here. There are several advantages of using variable stepsize method over the usual constant-stepsize formulae. It can be computationally advantageous to take the largest timestep that physically makes sense, and this value can change throughout a transient simulation. It also maintains the consistency of the approximations, unlike using a fixed-stepsize formula in a variable time-step problem. Using the coefficients $\{a\}$ of (A.8), (A.10), (A.12), or (A.14) the predictor of order m becomes

$$\mathbf{y}^{(*)} = \mathbf{y}^{(n)} + \Delta t \sum_{i=1}^{m} a_i \mathbf{g}\left(\mathbf{y}^{(n+1-i)}, t_{n+1-i}\right). \tag{4.18}$$

The corrector of order $m+1$, using (A.9), (A.11), (A.13), or (A.15) is

$$\mathbf{y}^{(n+1)} = \mathbf{y}^{(n)} + \Delta t \sum_{i=0}^{m} a_i \mathbf{g}\left(\mathbf{y}^{(n+1-i)}, t_{n+1-i}\right) . \tag{4.19}$$

The simplest technique to start the method is to begin with $m = 1$ for $t_0 \to t_1$ and increase m with each step until the desired order is reached.

4.3.1.2 As a predictor–corrector method in FSI

The use of the PE(CE)$^\infty$ method as the integrator for the structure directly mates with the *Structure predictor* and *Structure corrector* in Fig. 4.1. The structural model is integrated with a 4th order predictor and 5th order corrector.

A timestep begins with the predictor, (4.18), using the load information from the previous timestep. The kinematics of the particles are then computed and applied as boundary conditions to the fluid. After the fluid is advanced the updated fluid forces are applied to the body in the corrector equation (4.19), where the value of $\mathbf{y}^{(*)}$ is used as an estimate for $\mathbf{y}^{(n+1)}$. The check for convergence is based on the change between FSI iterations.

$$\text{error} = \left|{}_{(k+1)}\mathbf{y}^{(n+1)} - {}_{(k)}\mathbf{y}^{(n+1)}\right|_\infty \tag{4.20}$$

New kinematic information is applied to the fluid from the new values of ${}_{(k+1)}\mathbf{y}^{(n+1)}$. The structure continues in a fixed-point iteration, with counter k, using (4.19) to compute $\mathbf{y}^{(n+1)}$ until error $< \varepsilon$.

The benefits of this method are the ease and speed of computing a single iteration. The FEM assembly, implied in (4.17), is only on rank-1 arrays, since the mass and damping matrices are constant. Computationally it only requires level-2 BLAS for the matrix-vector multiplications, and a sparse solver to invert the symmetric $\mathbf{M}$ on a vector. However there is one significant downside to the use of

these methods that virtually prohibit their use in complicated FEM models. In the linear sense, the Adams methods must completely resolve all the frequencies of the structure. So as the spatial resolution increases so does the highest eigenvalue of the system. Therefore very careful attention to timestep size must be made when used in conjunction with CFD. Belytschko, Liu, and Moran (2000, Chap. 6) provide a way to estimate the structure's critical timestep. Since the body is nonlinear then this check represents a significant computational expense that needs to be performed periodically. For low Reynolds number flows around a body with many DOF, the limiting timestep will likely come from the body. In the parallel FLASH code this represents large inefficiency since it makes the entire simulation vastly more expensive. The Adams methods represent a very useful tool for models with few DOF, and as a way to check the results of other more complicated methods.

4.3.2 The Generalized-α method

The use of highly resolved finite elements results in both large storage requirements and also in severe timestep requirements for explicit integrators. Most finite element integrators use a variant of the seminal Newmark-β method (Newmark, 1959). The Generalized-α method (G-α) is a popular method in the dynamics of linear problems dating back to Chung and Hulbert (1993). It represents a unification of the methods of Hilber, Hughes, and Taylor (1977) and Wood, Bossak, and Zienkiewicz (1980), with improved characteristics. The G-α is second order in time, implicit, unconditionally stable for linear problems, and has user selectable dissipation of high frequencies. It was shown to be suitable for use for nonlinear problems in structural mechanics by Kuhl and co-workers (Kuhl and Ramm, 1996; Kuhl and Crisfield, 1999). The major consequences of using G-α on nonlinear structures is the loss of unconditional stability. A detailed analysis of the properties of the method for simple nonlinear systems are found in Baldo, Bonelli, Bursi, and Erlicher (2006); Bonelli,

Bursi, Erlicher, and Vulcan (2002), and Erlicher, Bonaventura, and Bursi (2002).

4.3.2.1 Formulation

The construction of the method makes two large assumptions: the kinematic relations of Newmark (1959), and the second order equation of motion

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{f}_{\text{int}}(\mathbf{q}) = \mathbf{h}(\mathbf{q}, t) \tag{4.21}$$

is approximated by the discrete equation

$$\mathbf{M}\ddot{\mathbf{q}}^{(\alpha_m)} + \mathbf{D}\dot{\mathbf{q}}^{(\alpha_f)} + \mathbf{f}_{\text{int}}^{(\alpha_f)} = \mathbf{h}^{(\alpha_f)}. \tag{4.22}$$

The superscripts imply the interpolation between time t_n and the next time step t_{n+1} or $t_n + \Delta t$.

$$\ddot{\mathbf{q}}^{(\alpha_m)} = (1 - \alpha_m)\ddot{\mathbf{q}}^{(n+1)} + \alpha_m \ddot{\mathbf{q}}^{(n)} \tag{4.23a}$$

$$\dot{\mathbf{q}}^{(\alpha_f)} = (1 - \alpha_f)\dot{\mathbf{q}}^{(n+1)} + \alpha_f \dot{\mathbf{q}}^{(n)} \tag{4.23b}$$

Kuhl and Ramm (1996) demonstrated the suitability of assuming that

$$\mathbf{f}_{\text{int}}^{(\alpha_f)} = (1 - \alpha_f)\mathbf{f}_{\text{int}}\left(\mathbf{q}^{(n+1)}, t_{n+1}\right) + \alpha_f \mathbf{f}_{\text{int}}\left(\mathbf{q}^{(n)}, t_n\right) \tag{4.24}$$

as a possible interpretation of the internal forces. Another possible view is to use $\mathbf{q}^{(\alpha_f)}$ directly. However this poses some technical difficulties when finding solutions and is not used here. Problems involving the prescribed motion of boundaries, such as flapping kinematics, require some special treatment which can be included by

augmenting the forcing term in a similar manner to the Adams formulation.

$$\begin{aligned}\mathbf{h}^{(\alpha_f)} = (1-\alpha_f)\mathbf{f}_{\text{ext}}\left(\mathbf{q}^{(n+1)}, t_{n+1}\right) + \alpha_f \mathbf{f}_{\text{ext}}\left(\mathbf{q}^{(n)}, t_n\right) \\ - \mathbf{M}_v\left((1-\alpha_f)\dot{\mathbf{v}}^{(n+1)} + \alpha_f \dot{\mathbf{v}}^{(n)}\right) - \mathbf{D}_v\left((1-\alpha_f)\dot{\mathbf{v}}^{(n+1)} + \alpha_f \dot{\mathbf{v}}^{(n)}\right)\end{aligned} \tag{4.25}$$

The classical Newmark equations

$$\dot{\mathbf{q}}^{(n+1)} = \mathbf{q}^{(n)} + \Delta t\left((1-\gamma)\ddot{\mathbf{q}}^{(n)} + \gamma\ddot{\mathbf{q}}^{(n+1)}\right) \tag{4.26a}$$

$$\mathbf{q}^{(n+1)} = \mathbf{q}^{(n)} + \Delta t\dot{\mathbf{q}}^{(n)} + \Delta t^2\left((1/2-\beta)\ddot{\mathbf{q}}^{(n)} + \beta\ddot{\mathbf{q}}^{(n+1)}\right) \tag{4.26b}$$

are then used with the relations above to cast (4.22) as a nonlinear algebraic equation with unknowns $\mathbf{q}^{(n+1)}$. Using the definitions in (4.23) and (4.26), expressions for $\dot{\mathbf{q}}^{(n+1)}$ and $\ddot{\mathbf{q}}^{(n+1)}$ are constructed so that only quantity reliant on time $n+1$ is $\mathbf{q}^{(n+1)}$.

$$\dot{\mathbf{q}}^{(n+1)} = \frac{\gamma}{\beta\Delta t}\left(\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) - \frac{\gamma-\beta}{\beta}\dot{\mathbf{q}}^{(n)} - \frac{\gamma-2\beta}{2\beta}\ddot{\mathbf{q}}^{(n)} \tag{4.27a}$$

$$\ddot{\mathbf{q}}^{(n+1)} = \frac{1}{\beta\Delta t^2}\left(\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) - \frac{1}{\beta\Delta t}\dot{\mathbf{q}}^{(n)} - \frac{1-2\beta}{2\beta}\ddot{\mathbf{q}}^{(n)} \tag{4.27b}$$

Chung and Hulbert (1993) parameterized all the constants of the G-α in terms of the spectral radius $\rho_\infty \in [0,1]$. This spectral radius asymptotically controls the annihilation of high frequencies in the response. It is this formulation that allows for minimal low frequency dissipation and maximal high frequency attenuation. Several other integration methods can be recovered by different constructions of the algorithmic parameters (Kuhl and Crisfield, 1999), these relations are summarized in Table 4.1. The flexibility to code a single implementation of G-α and have access to a wide range of methods merely by changing some constants represents a powerful incentive for the choice of G-α.

Table 4.1: Variants of the Generalized-α method in terms of the spectral radius ρ_∞ (Kuhl and Crisfield, 1999)

Algorithm	α_m	α_f	β	γ
Trapezoidal Rule	0	0	1/4	1/2
Newmark-β Method	0	0	$\frac{1}{(\rho_\infty+1)^2}$	$\frac{3-\rho_\infty}{2\rho_\infty+2}$
HHT-α Method	0	$\frac{1-\rho_\infty}{\rho_\infty+1}$	$\frac{1}{4}(1+\alpha_f)^2$	$\frac{1}{2}+\alpha_f$
WBZ-α Method	$\frac{\rho_\infty-1}{\rho_\infty+1}$	0	$\frac{1}{4}(1-\alpha_m)^2$	$\frac{1}{2}-\alpha_m$
Generalized-α Method	$\frac{2\rho_\infty-1}{\rho_\infty+1}$	$\frac{\rho_\infty}{\rho_\infty+1}$	$\frac{1}{4}(1-\alpha_m+\alpha_f)^2$	$\frac{1}{2}-\alpha_m+\alpha_f$

Combining and rearranging (4.22) through (4.27) gives the residual equation

$$\begin{aligned}\mathbf{0} = \mathbf{g}\left(\mathbf{q}^{(n+1)}\right) := {} & (1-\alpha_f)\mathbf{f}_{\text{int}}\left(\mathbf{q}^{(n+1)}, t_{n+1}\right) + \alpha_f \mathbf{f}_{\text{int}}\left(\mathbf{q}^{(n)}, t_n\right) \\ & - (1-\alpha_f)\mathbf{f}_{\text{ext}}\left(\mathbf{q}^{(n+1)}, t_{n+1}\right) - \alpha_f \mathbf{f}_{\text{ext}}\left(\mathbf{q}^{(n)}, t_n\right) \\ & + \mathbf{M}\left(\frac{1-\alpha_m}{\beta\Delta t^2}\left(\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) - \frac{1-\alpha_m}{\beta\Delta t}\dot{\mathbf{q}}^{(n)} - \frac{1-\alpha_m-2\beta}{2\beta}\ddot{\mathbf{q}}^{(n)}\right) \\ & + \mathbf{D}\left(\frac{(1-\alpha_f)\gamma}{\beta\Delta t}\left(\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) - \frac{(1-\alpha_f)\gamma-\beta}{\beta}\dot{\mathbf{q}}^{(n)} - \frac{(\gamma-2\beta)(1-\alpha_f)}{2\beta}\Delta t\ddot{\mathbf{q}}^{(n)}\right) \\ & - \mathbf{M}_v\left((1-\alpha_f)\ddot{\mathbf{v}}^{(n+1)} + \alpha_f\ddot{\mathbf{v}}^{(n)}\right) - \mathbf{D}_v\left((1-\alpha_f)\dot{\mathbf{v}}^{(n+1)} + \alpha_f\dot{\mathbf{v}}^{(n)}\right). \quad (4.28)\end{aligned}$$

Solving this equation is performed by Newton-Raphson iteration. Here is where the linearization of $\mathbf{f}_{\text{int}}$ comes into play. Using a Total-Lagrangian form for the equations of motion in a fixed inertial frame has the advantage of making $\mathbf{M}$ constant. For simple damping, such as a proportional damping model, $\mathbf{D}$ can also be assumed to be constant. This leaves only $\mathbf{f}_{\text{int}}$ and $\mathbf{f}_{\text{ext}}$ that need to be reassembled at each subiteration. Let the incremental displacements be

$$^{(i+1)}\mathbf{q}^{(n+1)} = {}^{(i)}\mathbf{q}^{(n+1)} + \Delta\mathbf{q} \quad (4.29)$$

where the (i) indicate the subiteration number of the Newton-Raphson method.

Linearizing (4.28) provides a linear equation for $\Delta\mathbf{q}$.

$$\left.\frac{\partial \mathbf{g}}{\partial \mathbf{q}^{(n+1)}}\right|_{{}^{(i)}\mathbf{q}^{(n+1)}} \Delta\mathbf{q} = -\mathbf{g}\left({}^{(i)}\mathbf{q}^{(n+1)}\right) \tag{4.30}$$

The effective tangent stiffness is

$${}^{(i)}\mathbf{K}_T := \left.\frac{\partial \mathbf{g}}{\partial \mathbf{q}^{(n+1)}}\right|_{{}^{(i)}\mathbf{q}^{(n+1)}} = (1-\alpha_f){}^{(i)}\mathbf{K} + \frac{1-\alpha_m}{\beta\Delta t^2}\mathbf{M} + \frac{(1-\alpha_f)\gamma}{\beta\Delta t}\mathbf{D} \tag{4.31}$$

where ${}^{(i)}\mathbf{K} = \frac{\partial \mathbf{f}_{\text{int}}}{\partial \mathbf{q}^{(n+1)}}$ is the geometrically exact stiffness matrix from the internal forces. There is no need to move a partition of the stiffness matrix to the right hand side of (4.30) to enforce a boundary condition. This information is already contained inside $\mathbf{f}_{\text{int}}$ on the right hand side. The symmetry of ${}^{(i)}\mathbf{K}_T$ is ensured unless follower-tractions on the surface of the body are included in the linearization. The construction of the initial condition on acceleration $\ddot{\mathbf{q}}^{(0)}$ is computed by solving (4.21).

$$\mathbf{M}\ddot{\mathbf{q}}^{(0)} = \mathbf{f}_{\text{ext}}\left(\mathbf{q}^{(0)}, 0\right) - \mathbf{f}_{\text{int}}\left(\mathbf{q}^{(0)}\right) - \mathbf{D}\dot{\mathbf{q}}^{(0)} - \mathbf{M}_v\ddot{\mathbf{v}}^{(0)} - \mathbf{D}_v\dot{\mathbf{v}}^{(0)} \tag{4.32}$$

The overall method to advance the structure a single time step is shown in Algorithm 4. The exit criterion of the Newton-Raphson iteration is based on the change in velocity between substeps. This error measure is used since the velocity of the surface is directly related to the calculation of pressure in the fluid. This expression can be simplified by using the definition from (4.27),

$$\begin{aligned}
{}^{(i+1)}\dot{\mathbf{q}}^{(n+1)} - {}^{(i)}\dot{\mathbf{q}}^{(n+1)} &= \frac{\gamma}{\beta\Delta t}\left({}^{(i+1)}\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) - \frac{\gamma}{\beta\Delta t}\left({}^{(i)}\mathbf{q}^{(n+1)} - \mathbf{q}^{(n)}\right) \\
&= \frac{\gamma}{\beta\Delta t}\Delta\mathbf{q}
\end{aligned}$$

Algorithm 4 The Generalized-α Method for a single time step $n \to n+1$

Precondition: ε, Δt, t_n, $\mathbf{q}^{(n)}$, $\dot{\mathbf{q}}^{(n)}$, $\ddot{\mathbf{q}}^{(n)}$, $\mathbf{f}_{\text{ext}}\left(t_{n+1}\right)$, and pre-computed constants

procedure SM_GENALPHA_ADVANCE
  Assume ${}^{(0)}\mathbf{q}^{(n+1)} = \mathbf{q}^{(n)}$
  $i \leftarrow 0$
  Prescribe kinematics $\mathbf{v}^{(n+1)}$, $\dot{\mathbf{v}}^{(n+1)}$, $\ddot{\mathbf{v}}^{(n+1)}$
  Compute $\mathbf{f}_{\text{ext}}\left({}_{(k)}\mathbf{q}^{(n+1)}, t_{n+1}\right)$ ▷ Includes (k) fluid loading
  while error $\leq \varepsilon$ **do**
    Assemble ${}^{(i)}\mathbf{K}_T$ ▷ (4.31)
    Assemble $-\mathbf{g}\left({}^{(i)}\mathbf{q}^{(n+1)}\right)$ ▷ (4.28)
    Solve for $\Delta\mathbf{q}$ ▷ (4.30)
    Update ${}^{(i+1)}\mathbf{q}^{(n+1)}$ ▷ (4.29)
    Update ${}^{(i+1)}\dot{\mathbf{q}}^{(n+1)}$ and ${}^{(i+1)}\ddot{\mathbf{q}}^{(n+1)}$ ▷ (4.27)
    Relative change in velocity ▷ (4.33)
    $i \leftarrow i+1$
  end while
  return $\mathbf{q}^{(n+1)}$, $\dot{\mathbf{q}}^{(n+1)}$, $\ddot{\mathbf{q}}^{(n+1)}$
end procedure

resulting in the normalized error expression

$$\text{error} = \frac{\gamma}{u_{\text{ref}}\beta\Delta t}\left|\Delta\mathbf{q}\right|_\infty . \tag{4.33}$$

4.3.2.2 As a predictor–corrector method in FSI

The G-α method is easily adapted for use in the *Structure predictor* and *Structure corrector* roles of Fig. 4.1 with some careful tweaking. There are three main points where discretion is required to use the method efficiently and robustly:

1. The choice of spectral radius ρ_∞.

2. The choice of ε in Alg. 4, and how it should be different between a predictor step and a corrector step.

3. The choice of FSI convergence criterion.

The choice of the spectral radius is a body specific problem. Since the radius is relative to the timestep a choice of a smaller timestep, say for the CFL condition of the fluid, would result in more temporal resolution in the structure. Therefore knowing estimates of the timestep that complies with the CFL, and knowing how many modes of the body are likely to be of interest provides an estimate for the value of ρ_∞. The bodies considered here, such as flapping wings or moving plates, are mostly undergoing bending deformation similar to their first mode shape. The CFL condition at low Reynolds numbers has a much smaller Δt requirement over the coarse FEM body. Very little excitation of the higher modes are seen, so a relatively small value of ρ_∞ is suitable for most of these types of problems.

Changing the value of ε between a prediction step and a correction step was found to be very beneficial to the FSI convergence rate. The first method explored set the predictor and corrector to only take a single Newton-Raphson step which resulted in FSI substeps in the 40-60 range. Upon a critical review of how the Newton-Raphson iterations take the body from time n to $n+1$ provides the basis for improving on that scheme. For bending problems, the first step of a Newton-Raphson method moves the body in the direction of bending; this makes sense since that is the direction with the lowest stiffness. The subsequent Newton-Raphson shifts the body axially. That first step commonly overestimates the displacement, and the subsequent iterations could be seen as pulling the body back to equilibrium. When the single Newton-Raphson step was used, the convergence rate was unpractical since the fluid was reacting to a body that was not close to equilibrium. Using these heuristics, the Newton-Raphson exit criterion ε can be intelligently changed.

- *Predictor*: Set the value of ε to 1×10^{-9}. This makes the predicted deformation field of the body relatively close to the previous deformation.

- *Corrector*: Set the value of ε to 1×10^{-3}. This allows the Newton-Raphson several substeps before returning to the fluid for an updated set of loads. The overprediction of a single Newton-Raphson step is avoided, and the fluid loads are very current. The structural solver does not waste time being very exact since the fluid loads will change.

The FSI convergence criterion must be set such that the change of the states during a *correction* are less than the tolerance ε_{FSI}, i.e. $\text{error}_{\text{FSI}} \leq \varepsilon_{\text{FSI}}$ where the error is defined as

$$\text{error}_{\text{FSI}} = \left| {}^{(0)}\dot{\mathbf{q}}^{(n+1)} - \dot{\mathbf{q}}^{(n+1)} \right|_{\infty}. \tag{4.34}$$

In light of how ε for the corrector is chosen, the condition on FSI must be more strict, $\varepsilon_{\text{FSI}} < \varepsilon$. Therefore a value of $\varepsilon_{\text{FSI}} = 1 \times 10^{-8}$ is chosen as the default. Also this scheme implies that that at least one correction step will always occur which ensures the strong coupling of the equations.

During test simulations with the values of ε as stated above, the number of FSI substeps dropped from 40-60 down to 5-8 for the same problem setup. This represents an incredible speed up and opens the possibility of using the method in production simulations.

Chapter 5

Numerical studies for three-dimensional cases

5.1 Dry numerical examples

Simple tests were used to test the implementation of the FEM model. These were designed to test the computation of internal forces, time marching, and moving boundary conditions. Each of these tests builds on the complexity of the numeric implementation: numerical integration, assembly, linear solution solving, and various boundary conditions. This testing process provided ample opportunity for efficiency increases and debugging before the solid model code was merged into the fluid code.

5.1.1 Static loading

Static load stepping is a standard continuation method, where an external load is increased while the body is kept in static equilibrium. Here, a full Newton-Raphson iteration is used to locate the equilibrium solution at each loading step. The simplest demonstration is the extension of an axial bar. The bar is fixed at one end, while on the other tip a force is applied. The size of the bar is $(l,\, w,\, h) = (30,\, 1,\, 1)$. The length was chosen to make the beam slender, and reduce the effects of the ends of the bar per Saint-Venant's Principle. This loading is not pure tension since the area of restrained face is constant. A pure tension case is feasible, but not interesting since a single finite element would reproduce the deformation field exactly. The mesh is comprised of uniformly sized elements with equal aspect ratios to keep the convergence properties well behaved. A sample mesh with 30 elements is shown in Fig. 5.1.

For a linear material in ideal tension, the load is related to the tip displacement

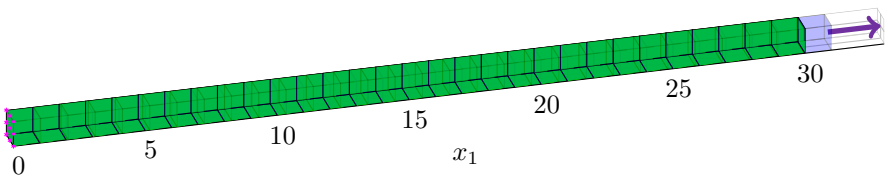


Figure 5.1: Example mesh with 30 elements for an axially loaded bar. Green is the initial configuration of the bar, and blue is the deformed bar.

δ by the well-known

$$\frac{P}{EA} = \frac{\delta}{l}\,. \tag{5.1}$$

Farahani and Bahai (2004) provide an analytic expression for an axially loaded rod for a wide range of material models. The Biot material response is identical to (5.1), while the Kirchhoff material can be expressed as

$$\frac{P}{EA} = \frac{1}{2}\left(\frac{\delta + l}{l}\right)\left[\left(\frac{\delta + l}{l}\right)^2 - 1\right]\,. \tag{5.2}$$

The results of increasing the load P on a uniform mesh of 30 quadratic elements is shown in Fig. 5.2. It appears as though the numerical implementation agrees very well with the analytic prediction.

The test of spatial convergence was performed by keeping the maximum load fixed to $\frac{P}{EA} = 1/30$, while increasing the number of elements. Using a relative error measure of the tip displacement the graph of Fig 5.3 was constructed. The reference displacement used was one mesh refinement greater than those shown in the figure. The rate of convergence for the tension test appears to be nearly quadratic. The displacement field is very simple, and the elements used here are quadratic, so the characteristics of Fig. 5.3 are expected.

A patch-test of sorts was designed for a more complicated loading condition. The simple shear of an isotropic elastic rectangular block is an example where the

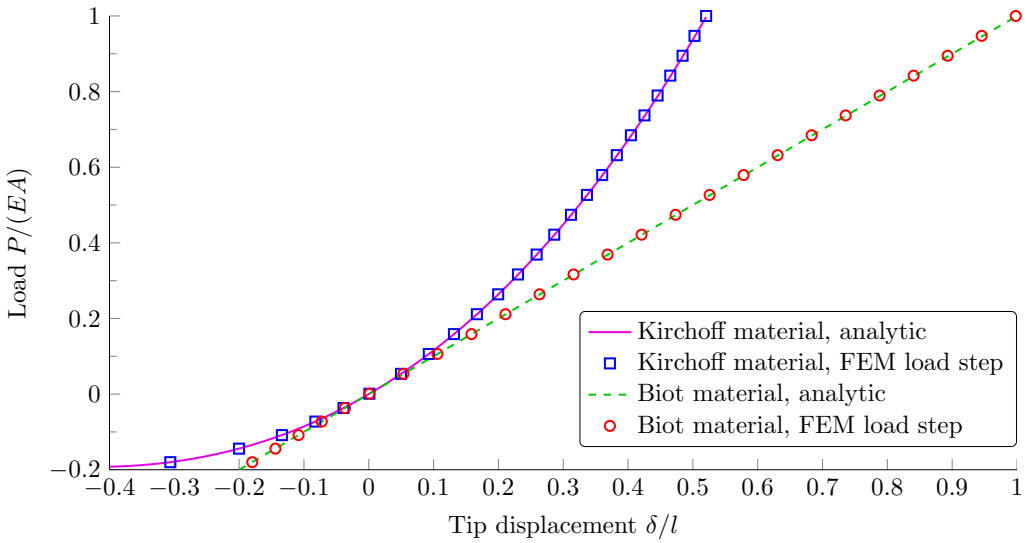


Figure 5.2: Tension and compression load-stepping results for the same axially loaded bar as Fig. 5.1. Also, the analytic values are shown for comparison.

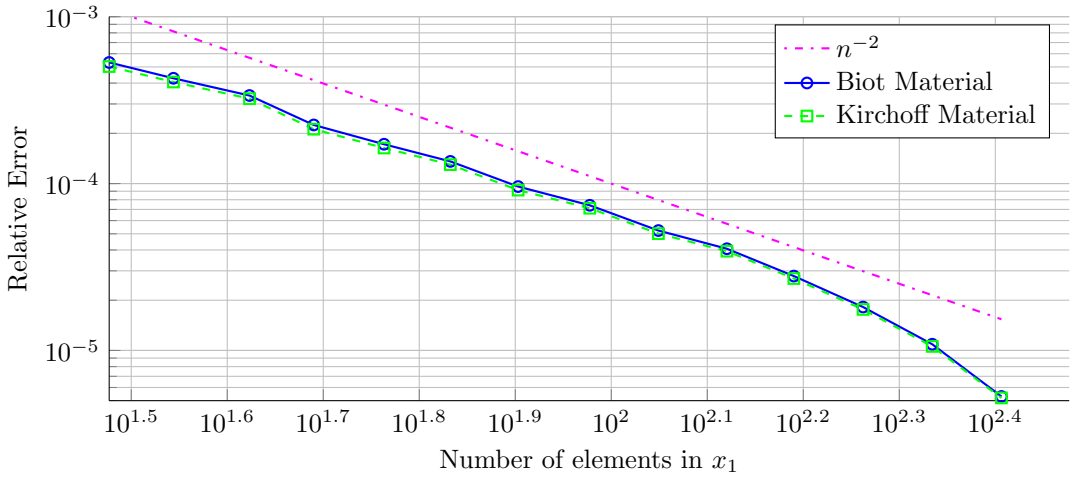


Figure 5.3: Convergence characteristics for the axially loaded bar with quadratic finite elements for Biot and Kirchhoff material models.

deformation and stress fields can be known analytically. This model problem is known as a *universal deformation* since it can be achieved by applying only surface tractions (Smith, 1993). Simple torsion by contrast cannot be achieved in finite deformation with only surface tractions. When using an irregular mesh, the test will demonstrate that the FEM implementation properly computes the displacement field. The surface tractions required to produce the simple shear deformation are derived in §A.4.4. The volume of the body is initially $[0, L] \times [0, L] \times [0, L]$, and is discretized with 8 elements. Before the loading stepping is performed, the nodal locations are randomly shifted to make sure that the isoparametric element implementation is working correctly. The outer surface of the block remains the same, but every node except the corners is moved a random amount. Then the body is fixed on the bottom surface, and the consistent surface loading is applied in increments of the amount of desired displacement k of the top surface. An example of the deformed body before and after the load stepping is shown in Fig. 5.4. Note that the mesh is intentionally of very low quality, but that the solution is still exact to machine precision.

Varying the load as a function of k shows good agreement between the theoretical model at the FEM implementation. In Fig. 5.5, k is varied across an extremely large range, and the maximum shear stress $\tau_{\max}(\boldsymbol{\sigma})$ is plotted. The strain field is theoretically homogeneous, so the stress is likewise constant. The maximal value of shear stress was chosen for its generality, and the error between the analytic model and numeric implementation is consistent across any chosen measure. The results of these tests indicate that the elastostatic implementation of the Biot, as well as reference Kirchhoff, material models are correct. Also, the computation of surface tractions on the elements works well.

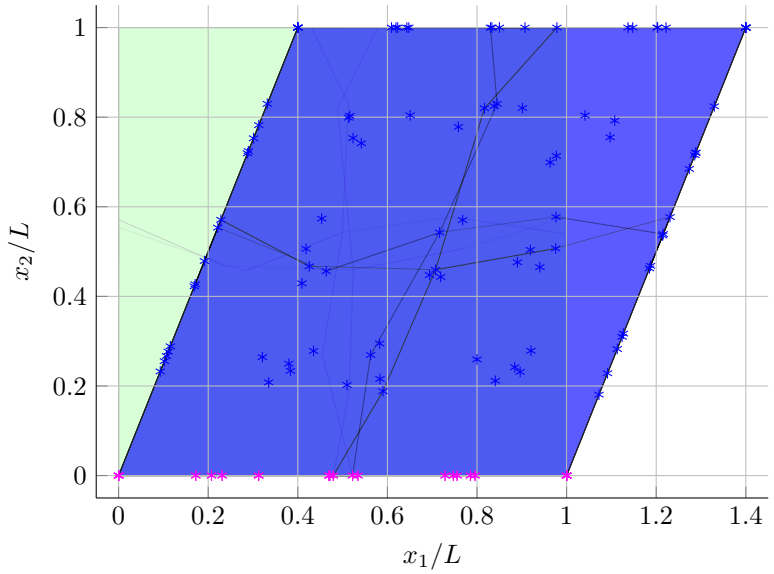


Figure 5.4: Example mesh from the simple shear test. The mesh is composed of 8 irregular elements that cover the $[0, L] \times [0, L] \times [0, L]$ domain. The green mesh is the initial configuration, and the blue configuration has been loaded with $k/L = 0.4$.

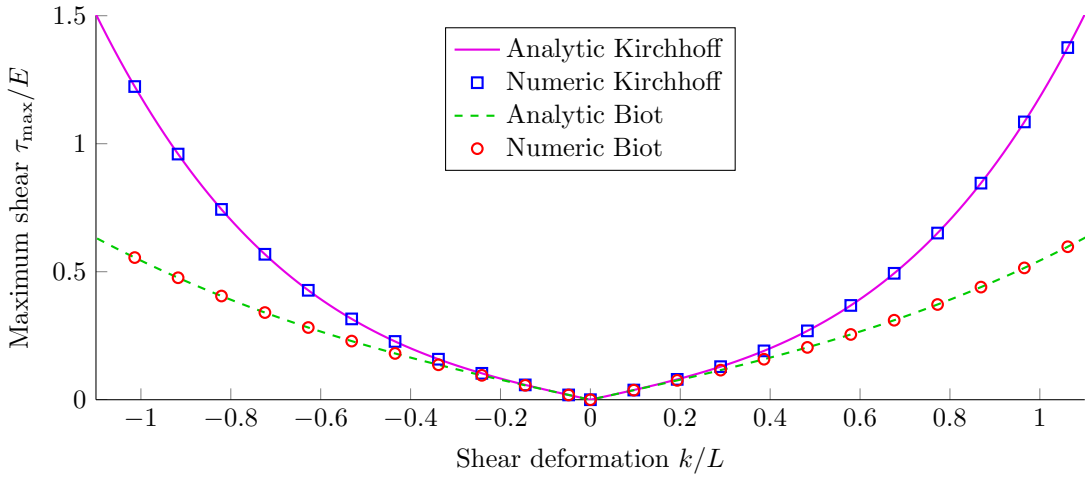


Figure 5.5: Maximum shear stress as a function of the shear deformation. The value of Poisson's ratio is $\nu = 0.3$.

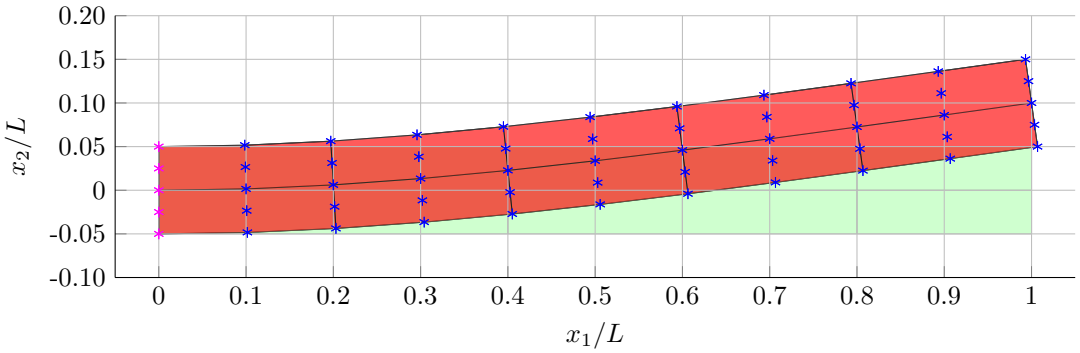


Figure 5.6: Solid model of beam with initial conditions set to the first eigenmode (scaled). Reference configuration in green, initial configuration in red, constrained nodes in magenta, unrestrained nodes in blue.

5.1.2 Dynamic response to initial conditions

A simple to test verify the response of the time integrator is to compute the free response to initial conditions. For this test a beam model was constructed out of quadratic hexahedral solid elements. The body's parameters were selected such that for the undeformed configuration the lowest natural frequency was 1 Hz. The geometry, and other material parameters are the same as the next example, shown in Table 5.1.

The initial displacements were chosen to be based on the first eigenmode of the clamped-free beam, as shown in Fig. 5.6. The eigenmode was scaled such that center of the free tip was placed at $x_2(0) = 0.1L$. This amount of initial displacement is on the upper edge of what linear models can predict and was chosen to see how the nonlinear FEM would respond. Since this shape elongates the body it introduces high frequency axial vibrations as well as the desired bending motion. This provides a platform to investigate how the integrators respond to a wide band of frequencies. In the linear model the vertical tip will oscillate as $x_2(0)\cos(2\pi t)$. The simulation was integrated using both the 4th order Adams-Bashfourth-Moulton predictor-corrector and G-α with two different values of spectral radius ρ_∞.

Numerical results of the implemented model are in Fig. 5.7. The top part

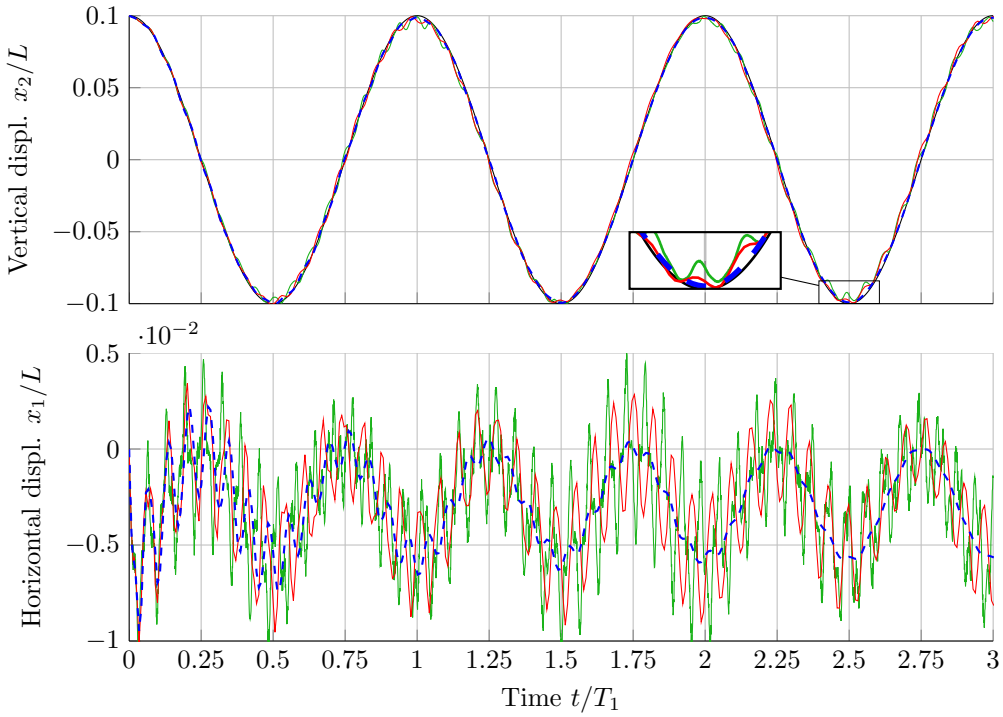


Figure 5.7: The position of the center face of the beam's tip. In green is the 4th order Adams-Basthfourth-Moulton predictor-corrector with $\Delta t/T_1 = 8 \times 10^{-5}$. The G-$\alpha$ with time step $\Delta t/T_1 = 0.008$ is in red for $\rho_\infty = 0.8$ and blue for $\rho_\infty = 0.4$.

of the figure shows the vertical displacement of the free tip's face, while the lower plot shows the horizontal displacement in time. For both the 4th order Adams predictor-corrector and the G-α method the vertical displacement is mostly the same. However, looking at the inset plot around $t/T_1 = 2.5$, higher frequencies of oscillation are present in both the Adams method and the G-α with $\rho_\infty = 0.8$. These modes correspond to frequencies imparted from the initial conditions causing elongation of the beam. Even though the Adams method is explicit the time step is 100 times smaller than the G-α which results in drastically more wall time for the same simulation time.

Looking at the lower plot in Fig. 5.7 reveals a more interesting story about how the different integrators are handling the various scales of oscillation. These results follow the theoretical predictions on each of the methods when applied to a

linear system, and it is comforting to see the same trends in the nonlinear models. It is clear that the Adams methods is resolving the highest possible frequency, which follows from the condition necessary for the stability of the predictor. There is some damping the Adams system however, since the amplitude is decreasing. Comparing the G-α results show that as $\rho_\infty \to 1$ the higher frequencies would be resolved. At the lower end of ρ_∞ however, that after $t/T_1 = 2$ all but the lowest frequencies have been attenuated. This is not damping in the classical sense, it is a visible manifestation of the low-pass filter of the G-α method. This demonstrates the applicability of G-α for large motions, when the higher frequency information is not important. This has the added benefit of transferring less noise to the fluid in a FSI situation.

5.1.3 Moving boundary response

To test if the implementation can handle the prescribed motion of a boundary a simple case was devised to spin a beam-like model as a rotor. The model is again comprised of 27-node solid elements and one surface is constrained to move. In order to make the implementation of these kinematics very general they were implemented using Rodrigues' rotation formula (Bauchau, 2011) about an arbitrary point, as illustrated in Fig. 5.8. The prescribed parameters are the angle $\theta(t)$, the axis of rotation unit vector $\hat{\boldsymbol{n}}$, and the location of the pivot $\boldsymbol{b}$.

The rotation matrix is constructed as

$$\mathbf{Q} = \mathbf{I} + \sin\theta[\hat{\mathbf{n}}]_\times + (1 - \cos\theta)(\hat{\mathbf{n}}\hat{\mathbf{n}}^T - \mathbf{I}). \tag{5.3}$$

Where the cross product identity has been used

$$[\hat{\mathbf{n}}]_\times := \begin{bmatrix} 0 & -n_1 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{bmatrix}.$$

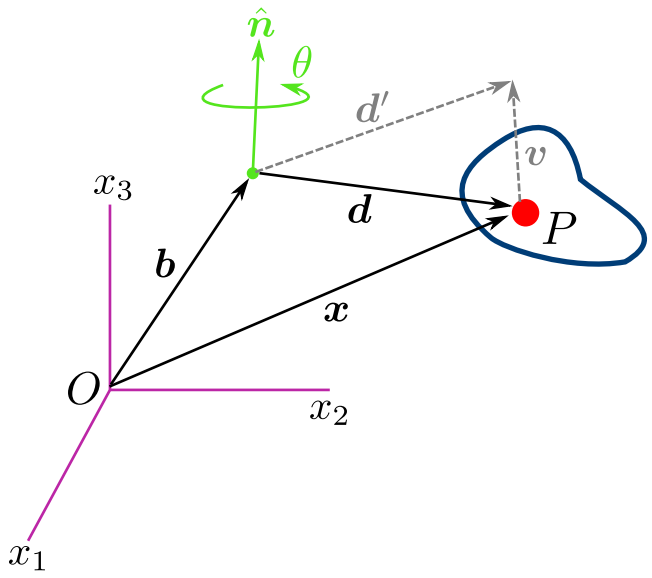


Figure 5.8: Diagram of the parameterization for prescribing the whirling motion of a point P about an arbitrary location using an axis of rotation $\hat{\boldsymbol{n}}$ and angle θ.

The rotated vector $\boldsymbol{d}' = \boldsymbol{Q}\boldsymbol{d}$ can be used with the construction of Fig. 5.8 to compute the displacement of point P.

$$\boldsymbol{d}' = \boldsymbol{Q}\boldsymbol{d}$$

$$(\boldsymbol{d} + \boldsymbol{v}) = \boldsymbol{Q}\boldsymbol{d}$$

$$\boldsymbol{v} = (\boldsymbol{Q} - \boldsymbol{I})\boldsymbol{d}$$

This gives the displacements being prescribed as

$$\boldsymbol{v} = (\boldsymbol{Q} - \boldsymbol{I})(\boldsymbol{x} - \boldsymbol{b}). \tag{5.4}$$

For the whirling case here, the angle is prescribed by

$$\theta(t) = (1 - e^{-t/\tau})\omega t \tag{5.5}$$

where τ damps the impulsive start, and ω is the target constant angular velocity.

Table 5.1: Parameter values for the whirling beam

Parameter	*Symbol*	*Value*	*Notes*
Size	$(l,\ w,\ h)$	$(1,\ 0.1,\ 0.1)$	
Density	ρ	10	
Young's Modulus	E	3.77×10^4	Makes the first nat. freq. 2π
Pivot	$\mathbf{b}$	$(0, 0, 0)$	
Rotation axis	$\hat{\mathbf{n}}$	$(1, 1, 1)/\sqrt{3}$	
Angular speed	ω	$2\pi/3$	Spins at 1/3 the first nat. freq.
Time const.	τ	0.1	

Values of the parameters of the model are shown in Table 5.1.

The results are post-processed in MATLAB, and the resulting composite figure is shown in Fig. 5.9. The results demonstrate that the implementation of a moving boundary is successful. Therefore this implementation allows for any $\mathcal{C}^2$ function of time to be used to arbitrarily move a boundary.

5.2 Wet examples

5.2.1 Moving plate

In order to test the FSI code a simple, and relatively small sized problem was required. Recently, Cleaver, Calderon, Wang, and Gursul (2013a); Cleaver, Wang, and Gursul (2013b) performed experiments on compliant plates in a water tunnel. The force measurements that this work provides could supply any practitioner of FSI numerics with a simple validation case. The problem setup here has been simplified to make the domain manageable for a small number of nodes on the HPC. Here, only 64-128 processors are used with wall times between 12 to 18 hours.

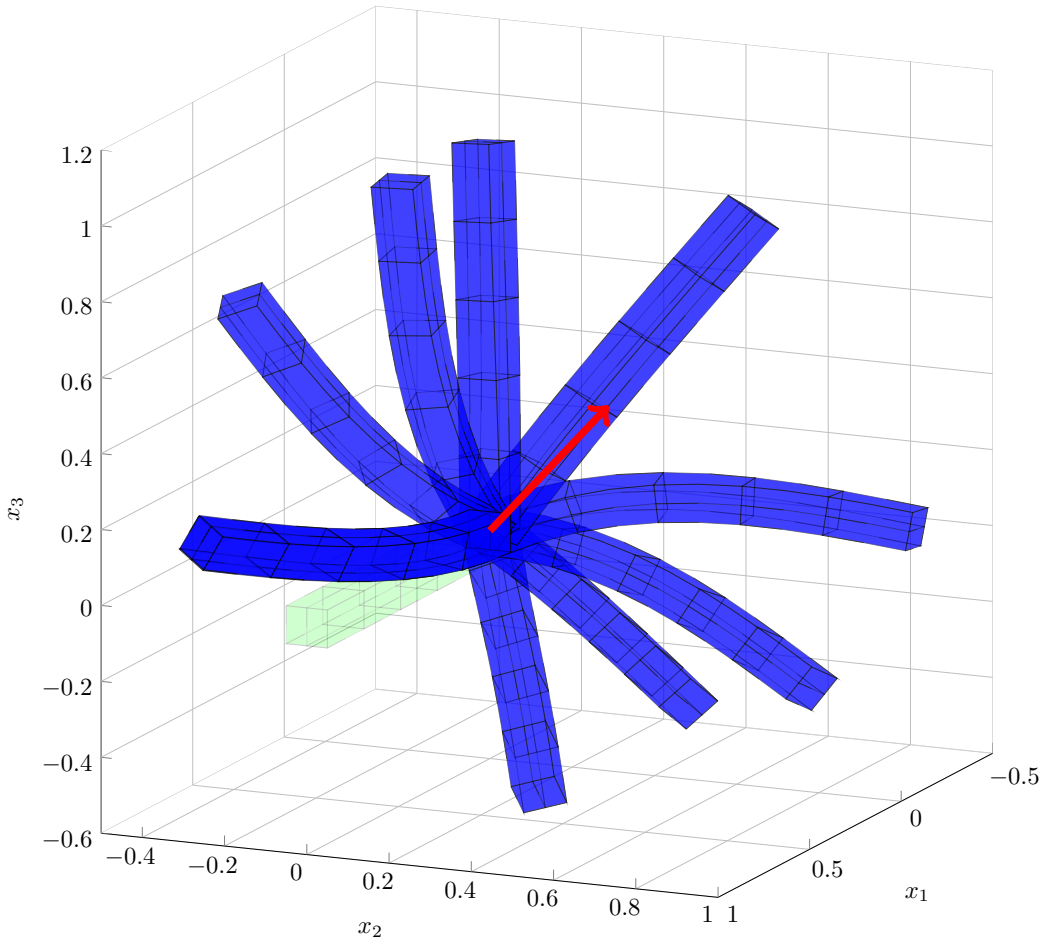


Figure 5.9: Beam model at equally spaced snapshots in time for the second revolution of the whirling. The green elements are the reference configuration, the blue elements are the deformed configuration in time, and the red arrow is the axis of rotation.

5.2.1.1 Geometry and kinematics

A moving plate of length L, width $0.3L$, and thickness $0.05L$ is centered in a $3L \times 3L \times 3L$ quiescent fluid domain. The plate is rotated 15° along its long axis. The density of the body is varied between $\rho/\rho_{\text{fluid}} \in \{1, 2, 10\}$. The displacements of one of the short edges is restrained by the prescribed kinematics

$$x_1(t) = 0 \tag{5.6a}$$

$$x_2(t) = 0 \tag{5.6b}$$

$$x_3(t) = \left(1 - e^{-t/\tau}\right) A_3 \sin\left(\omega_f t\right) \tag{5.6c}$$

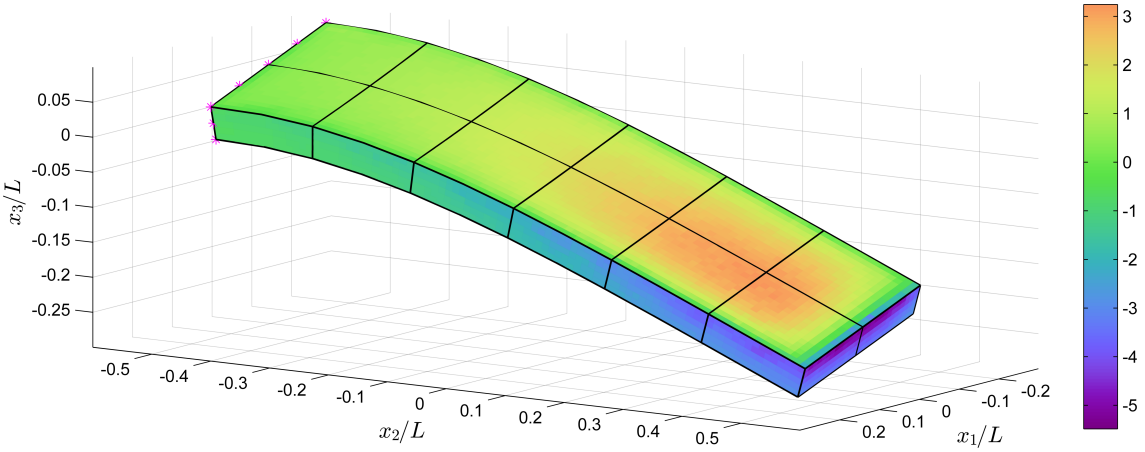
Where the time constant $\tau = 0.05$ is chosen as a small number to remove an impulsive start but not affect the kinematics for long. Poisson's ratio was chosen to be 0.3 and Young's modulus was chosen such that the first natural frequency of the body was $\omega_f/\omega_1 = 1/3$, see §A.3.5.1 for the procedure. The maximum prescribed velocity is chosen as the reference speed

$$u_{\text{ref}} = \max\left|\dot{x}_3(t)\right| = A_3\omega_f. \tag{5.7}$$

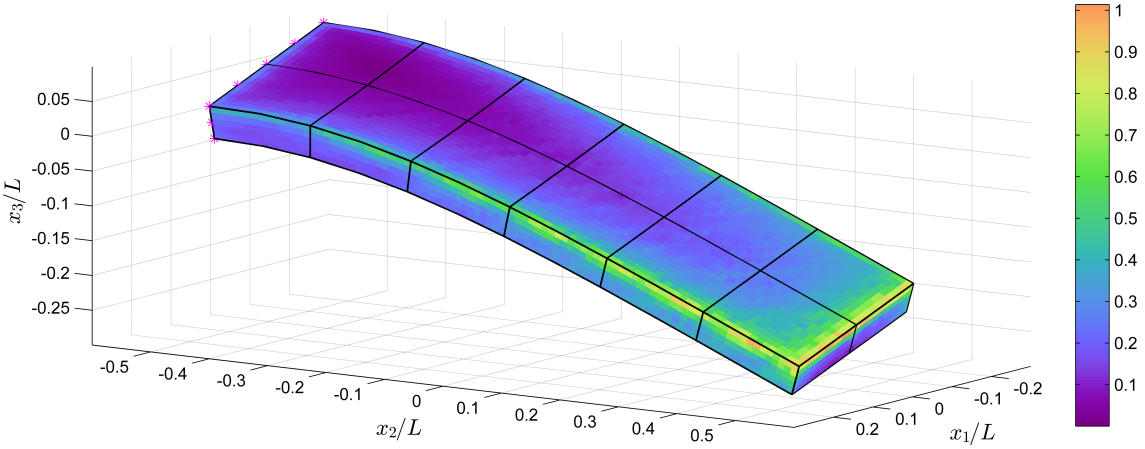
The amplitude of oscillation is set to $A_3 = 0.15L$, while the Reynolds number is constructed as

$$Re = \frac{\rho_{\text{fluid}} L u_{\text{ref}}}{\nu}\,. \tag{5.8}$$

These parameters are not the same used by Cleaver *et al.* (2013a,b), the Reynolds number has been lowered from 10 000, there is no free-stream velocity, and the domain is smaller. The setup tested here is merely a first step, and the use of future resources would allow for larger problems that exactly replicate the published work.



(a) Pressure on the body. Colors correspond $C_p/2$.



(b) Magnitude of the viscous stress. Colors correspond to $C_f/2$.

Figure 5.10: Example of the surface stresses acting on the body. These are the instantaneous values at $t/T_f = 2.05$, for $Re = 200$, $\rho/\rho_{\text{fluid}} = 1$.

5.2.1.2 Results

The solution is computed for several periods $T_f = 2\pi/\omega_f$ of forcing. Figure 5.10 shows the surface tractions on a body. From these plots it is clearly evident that even for $Re = 200$, the viscous stresses are mostly quite low, except at the edges of the body. The pressure is acting as we would expect with a large high pressure opposing the motion of the plate.

Computing the center of mass, as described in §A.3.3, shows how the effect of mass density changes the FSI. Figure 5.12 shows the center of mass for densities

$\rho/\rho_{\text{fluid}} \in \{10, 2, 1\}$ at $Re = 200$ as well as a dry body. The displacements in x_1 and x_2 appear to change drastically as the density ratio is lowered, and the FSI forces of the fluid begin to dominate the body. This is especially visible in the x_2 direction where the displacements are nearly 10 times those of the dry (non-FSI) case. The fluid also appears to be damping out the higher frequencies of the response. This is seen in the x_3 direction, where only the dry body appears to have multiple frequencies in the response.

Integrating the surface stresses across the body provides a way to compare the fluid contributions across the density range. The total force due to viscous stresses is quite small for all time in Fig. 5.12. This correlates well to the instantaneous field in 5.10b where the peaks may have been large, but they were highly localized. The pressure forces are dominating in all three directions by an order of magnitude.

The visualization of the flow fields of Fig. 5.13 were constructed using Tecplot 360$^{\text{TM}}$. At several instances of time, a number of isocontours of the Q-criterion (Hunt *et al.*, 1988) are plotted along with a x_2-x_3 slice of the x_3 velocity field. Depicted in the left column are results for $Re = 200$, and in the right column are $Re = 1000$ at nearly the same instances of time. The density ratio of the body is $\rho/\rho_{\text{fluid}} = 1$, which as demonstrated in the previous figures results in the largest deformation and highest loads. The Q-isosurfaces are not symmetric about the x_2-x_3 plane since the body is rotated along the x_2-axis. Comparing side-by-side frames shows that the flow structures appear smaller and remain longer as Reynolds number is increased. At $t/T_f = 1.61$ for $Re = 1000$ the Q-isosurface appears to be rolling up on itself in a hairpin–like manner (Bernard, 2011).

Overall, the results from these tests appear promising that the FSI algorithm works efficiently enough to make serious problems practical for calculations. This is assuming that the necessary HPC power is available.

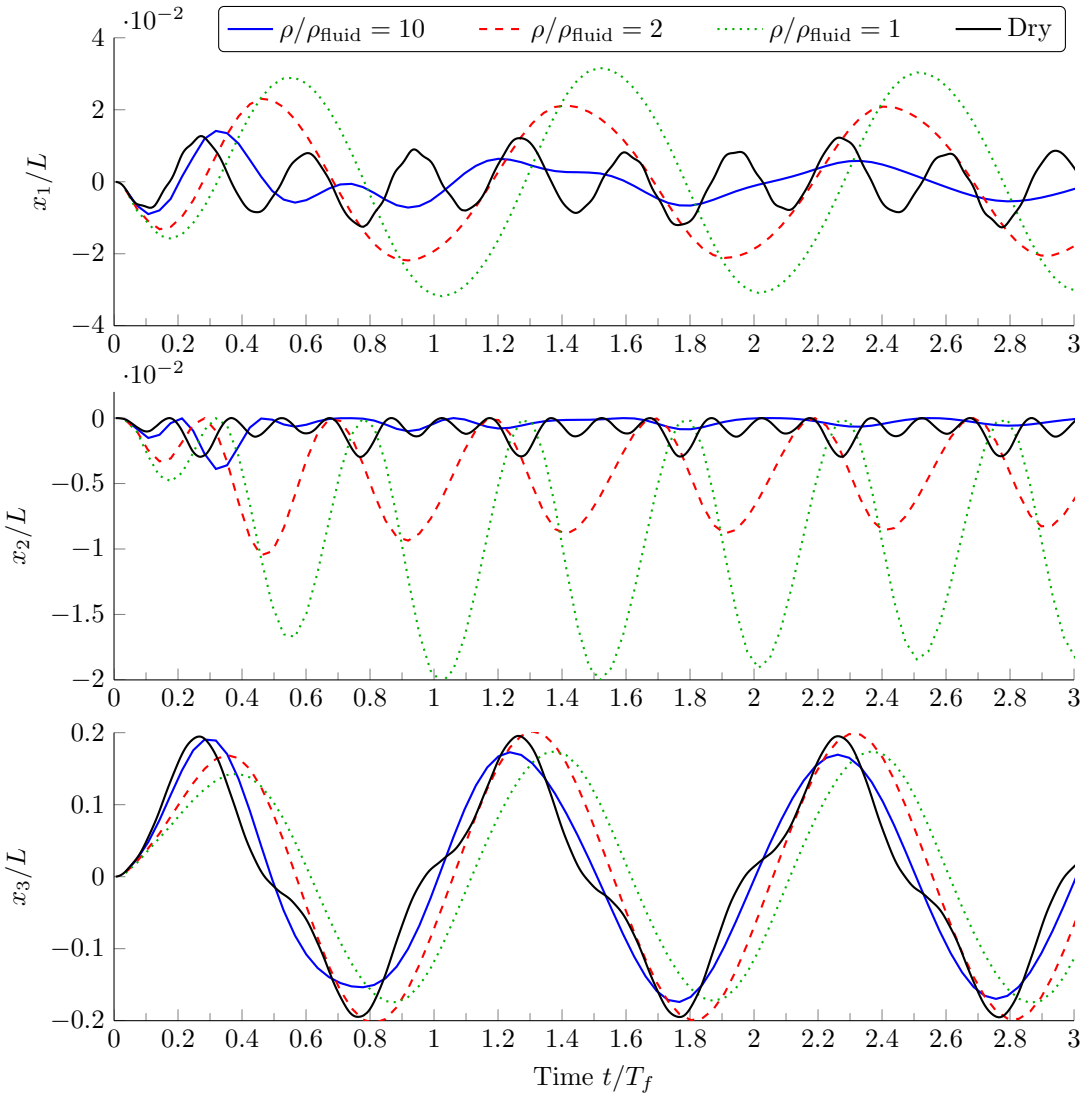


Figure 5.11: Position of the center of mass in the deformable plate for $Re = 200$.

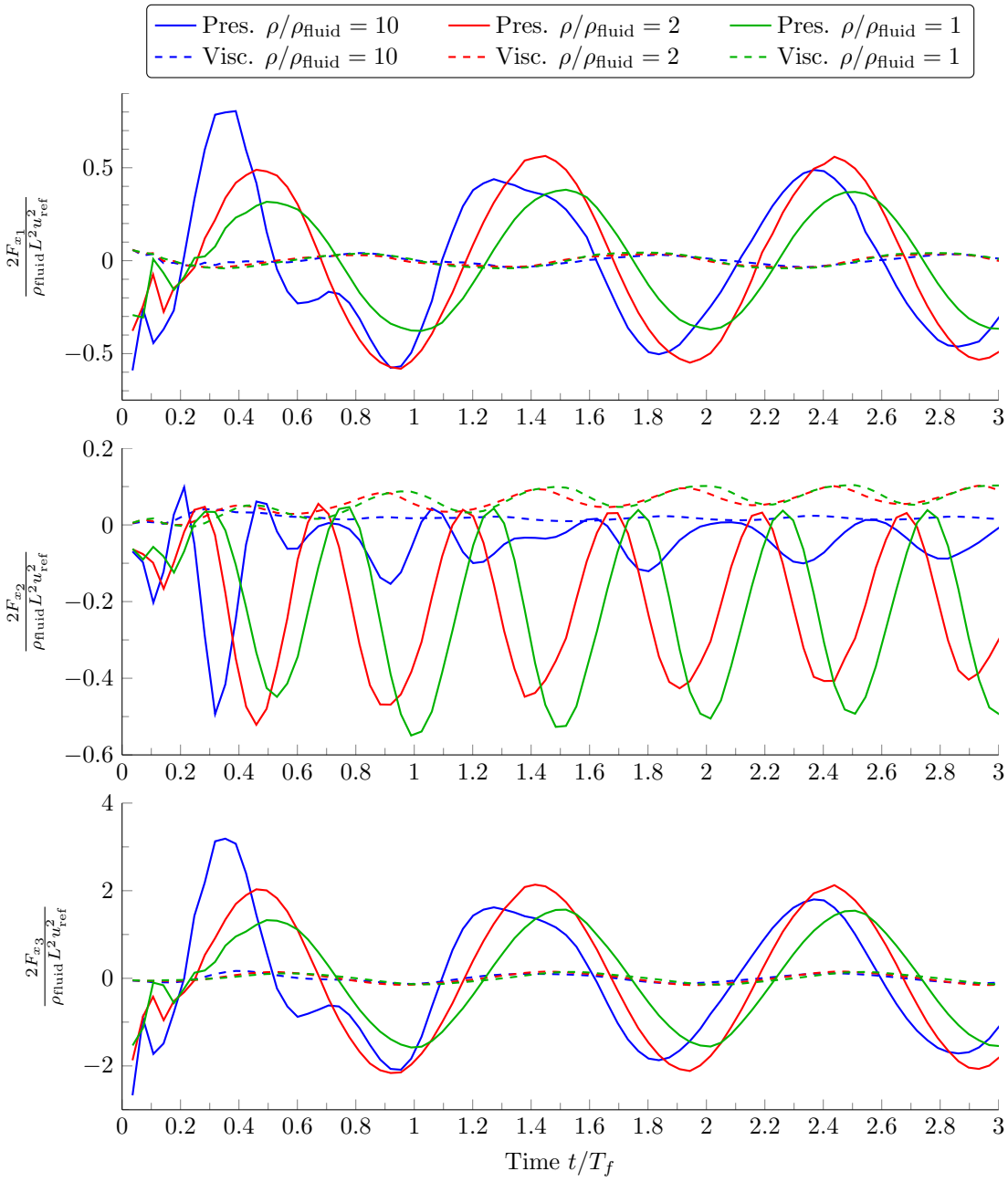


Figure 5.12: Fluid forces computed at the centroid of the deformable plate for $Re = 200$.

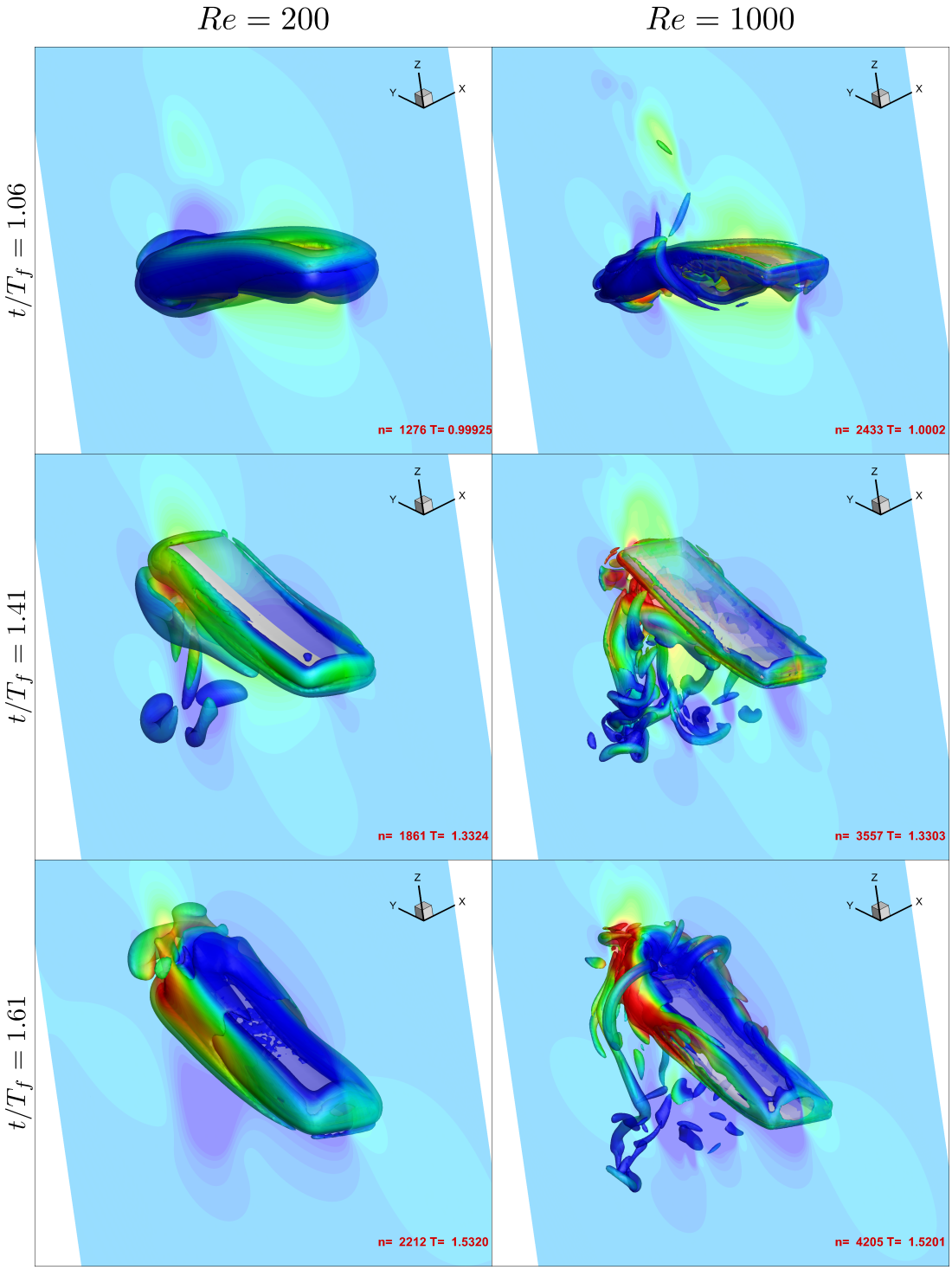


Figure 5.13: Demonstration of FSI for a flexible plate with $\rho/\rho_{\text{fluid}} = 1$, and $\omega_f/\omega_1 = 1/3$. The time has been nondimensionalized with T_f. Shown are isosurfaces of the Q-criterion and a slice of the z velocity in the y-z plane.

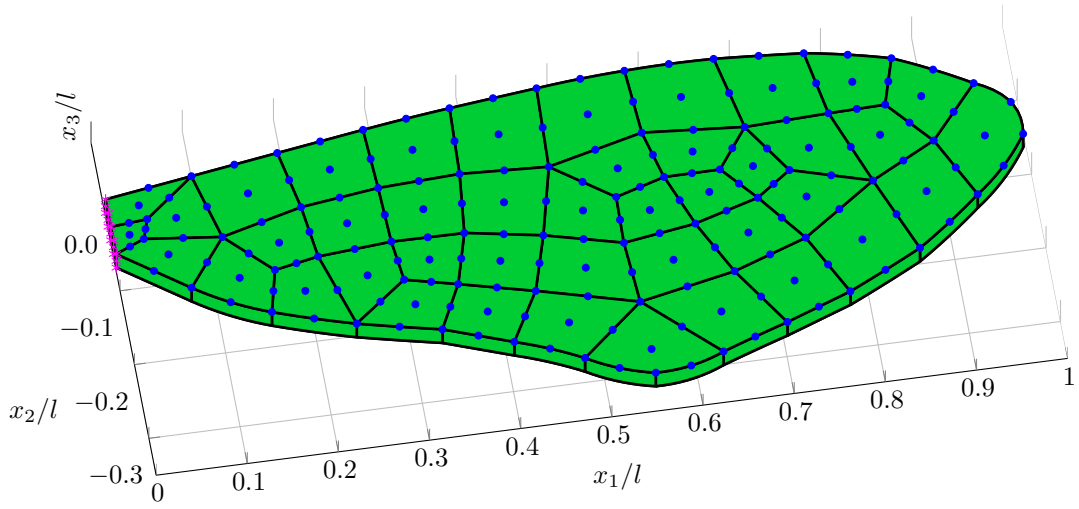


Figure 5.14: Mesh of a *Manduca sexta* inspired wing. The planform is based on the results of O'Hara and Palazotto (2012). The restrained nodes are at the root in magenta. The free nodes are blue (only the top surface is shown for clarity).

5.2.2 Flapping wing

This section discusses the setup of a flexible flapping wing. The final simulations have not yet been performed due to lack to time and processor power. However it is believed that from the results reported in the previous sections that the implementation of the FSI method would successfully compute the results.

5.2.2.1 Geometry and kinematics

The first insect inspired geometry to be modeled is the forewing of a *Manduca sexta*. Figure 5.14 shows the first draft of a 3D FEM mesh of the planform. This shape was extract from a figure in O'Hara and Palazotto (2012). Another useful reference for wing planforms are the many drawings in Comstock (1918), as shown for a *Musca domestica* wing in Fig. 5.15.

The driving kinematics are based on the functions of Berman and Wang (2007). This angular description of flapping motion is extremely versatile to a broad range of flapping styles. The construction of how these kinematics are applied to the FEM

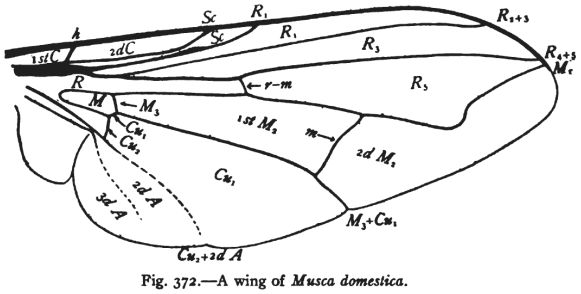


Fig. 372.—A wing of *Musca domestica*.

Figure 5.15: Detailed sketch of a *Musca domestica* wing from Comstock (1918). This present future possibilities both in terms of planform geometry and material distribution.

restrained surface is detailed in §A.5. The hawkmoth parameters of Berman and Wang (2007) indicate an almost constant angle of attack for the majority of the hover stroke with relatively fast reversals at the ends of the stroke. Preliminary dry tests with $\omega_f/\omega_n = 1/3$ for a homogeneous body indicate that the body is too soft. The most realistic correction is to stiffen the root and leading edge of the wing. This follows from the parameter distribution proposed by Combes and Daniel (2003b) as well as the mode shapes found in §3.1.2.

The Reynolds number of a *Manduca sexta* based on the span length and peak tip speed would be around 25×10^3 to 29×10^3. This increase in Reynolds number drastically changes the computing requirements for a practical simulation. If uniform grids are used for example, assuming that the cell size would need to approximately be $\Delta x = 0.002L$, then a *small* domain of $4L \times 4L \times 4L$ would have roughly 2048^3 points. If the FLASH block size is 32^3, this results in 64^3 blocks to distribute on the HPC. For these sized blocks, it was found that placing more than 30 blocks per processor results in poor scaling. Therefore 216 224 blocks at 30 blocks per processor results in 8739 processors. That many processors represent a major use of resources, and it is beyond the clusters currently located at the University of Maryland. The use of AMR and lower Reynolds number cases will need to be considered as the next practical steps.

Chapter 6

Summary and concluding remarks

In this work, new tools have been constructed to explore the complex physics of nonlinear fluid-structure interactions. Although the methods are presented towards the application of insect flapping, they are general and could be applied to a wide range of problems. The key aspects of this work are the following:

1. Analysis of 2D models of flexible flapping flight. Two different fluid models are compared to see under what conditions an inexpensive inviscid model is suitable. Several numerical tools, such as numerical continuation, dimension calculation, and the proper orthogonal decomposition are used to describe various aspects of the model and its results.

2. Experiments on living insects were conducted to extract modal information of the forewing. These are the first experiments of their kind performed on the attached wing of a living insect. The modal information gathered provides the first building block for a database on the spectral aspects of insect wings, and as a starting point for bio-inspired wing design.

3. The basic formulation of a FEM structural model that is geometrically exact and is capable of handling a variety of material behaviors is presented from fundamental continuum mechanics theory. This novel material law relies on fewer assumptions compared to most finite deformation laws and is arguably more physical than the Kirchhoff law. The linearization for implicit methods is compact and efficient. The law is formulated in several ways so that it can be directly included into most existing types of FEM frameworks.

4. A novel partitioned FSI algorithm using the Adams and Generalized-α methods

is described. This method is suitable for large scale systems, with bodies of tens of thousands of degrees of freedom. Special attention is given to the time step requirements, and how this method decouples the fluid grid generation from the body mesh generation. A consistent method to project the boundary conditions between the body and the fluid is constructed using the Lagrangian markers in the PARTICLES unit of FLASH.

5. Finally, numerical demonstrations of the above methods show the effectiveness of the algorithms and implementations. A fully flexible plate is excited in a flow over a range of densities and Reynolds numbers. It was found that the methods worked well, even for equal densities between the fluid and the solid. Large deformations were handled without issue by only 12 elements, and the FSI substeps were kept low (between 5-8).

There is a great deal of possible future directions for this research. In 2D, the use of optimization tools to exploit the computational speed of the UVLM would be an interesting path. The 2D DNS could be varied over more parameters such as including damping, making the structure asymmetric, or including alternate kinematics. All of these investigations would still involve only 2D physics which may or may not be relevant to evolving 3D designs.

The topic of flapping is still wide open in 3D. The codes introduced here, coupled with the necessary computational resources, could be used for a variety of studies. The first should be of the *Manduca sexta* wing. Transient phenomena like clap and fling would be a good use for two bodies in the fluid. If detailed CT-scans of wings were located, then future directions could include a study on how the local 3D geometry affects the flight performance compared to a simple geometry wing. The current code is implemented for multiple bodies of small to medium scale. If a body with hundreds of thousands of DOF was of interest then using a library like PETsc (Balay *et al.*, 2013) may be required.

Completely capturing the physics involved in the FSI of flapping wing systems is currently only possible with large scale simulations. These types of predictions are useful in the understanding of natural fliers like insects, and also in exploiting the nonlinear phenomena for engineering uses. The future of flapping MAVs depends heavily on battery capacity, and the optimization of designs. The tools presented here aid in uncovering aerodynamic efficiency, and could be used to search for new regions of parametric interest.

Appendix A

Notation, formulation, and implementation details

A.1 Notation

Since this work crosses the well established traditions of disciplines such as dynamics, linear algebra, continuum mechanics, and fluid dynamics the notation is a mess. Table A.1 is my attempt to write in a single style and is mostly consistent throughout the dissertation. This style is largely based on that of Antman (2004).

Table A.1: Notional conventions for math quantities

Quantity	**Group**	**Symbols**	**Notes**
Scalars	$\mathbb{R}$ or $\mathbb{Z}$	a, b, ... α, β, ...	italics
Vectors	$\mathbb{E}^3$	$\boldsymbol{a}$, $\boldsymbol{b}$, ...	Lower case, bold italics
2^{nd} Order Tensors	$\text{Lin} := \mathcal{L}(\mathbb{E}^3, \mathbb{E}^3)$	$\boldsymbol{A}$, $\boldsymbol{B}$, ... $\boldsymbol{\Lambda}$, $\boldsymbol{\Theta}$, ...	Upper case, bold italics
4^{nd} Order Tensors	$\mathcal{L}(\text{Lin}, \text{Lin})$	$\mathfrak{A}$, $\mathfrak{C}$, ...	Bold, Fraktur
n-Tuple	$\mathbb{R}^n$	a, b, ...	Lower case, bold, sans
Matrix	$\mathbb{R}^{n\times n}$	A, B, ... $\mathsf{\Lambda}$, $\mathsf{\Theta}$, ...	Upper case, bold, sans

A.2 Variable stepsize Adams methods

The classical linear-multistep methods of the Adams family are usually described in texts for a constant stepsize Δt. It can be advantageous in scientific computing to use the largest stepsize possible; one that is both physically and numerically acceptable. This is often expressed as the well-known Courant-Friedrichs-Lewy

(CFL) condition. At times, constant stepsize methods are used in varying timestep problems, usually with a decrease in the stability and a possible loss of accuracy. Commercial software such as the MATLAB routine `ode113` employs a rather sophisticated adaptive Adams method (Shampine and Reichelt, 1997). In the FLASH code the stepsize control is already in-place. It is not adaptive time marching, it only varies the timestep to the minimum allowed according to the response from each program Unit. Therefore what is needed is to build Adams methods in terms of old values of Δt. The goal is to find formula of the form

$$y_{n+1} = y_n + \Delta t \sum_{i=0}^{m} a_i f_{n+1-i} \tag{A.1}$$

where $\{a\}$ are functions of old timesteps, y_n is the state at t_n, and $f_n = f(y_n, t_n) = y'(t_n)$. For explicit methods (Adams-Bashfourth) $a_0 \equiv 0$ while for implicit methods (Adams-Moulton) $a_0 \neq 0$.

A.2.1 Explicit formulae

Remembering that the Adams methods are constructed by the integration of an interpolated function, Hairer, Nørsett, and Wanner (1993, III.5) outline a method for building variable timestep methods. First let's recall the recursive definition for divided differences.

$$\begin{aligned} f[t_n] &= f_n \\ f[t_n, \ldots, t_{n-j}] &= \frac{f[t_n, \ldots, t_{n-j+1}] - f[t_{n-1}, \ldots, t_{n-j}]}{t_n - t_{n-j}} \end{aligned} \tag{A.2}$$

Let $p(x)$ be the interpolated function

$$p(x) = \sum_{j=0}^{m-1} \prod_{i=0}^{j-1} (x - t_{n-i}) f[t_n, t_{n-1}, \ldots, t_{n-j}]. \tag{A.3}$$

Now the approximation to $y(t_{n+1})$ is defined by

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} p(x)\ dx\,. \tag{A.4}$$

Hairer *et al.* (1993) has some alternate ways to make the integration of $p(x)$ a little simpler but they are not needed for the simple cases considered here. For an example calculation to build an explicit method of order 2 (i.e. $k = 2$) we have

$$p(x) = f_n + \frac{(x - t_n)}{t_n - t_{n-1}}\,(f_n - f_{n-1})\,.$$

Placing this inside (A.4) and evaluating the integral gives

$$y_{n+1} = y_n + (t_{n+1} - t_n)f_n + \frac{(t_{n+1} - t_n)^{\,2}}{2\,(t_n - t_{n-1})}\,(f_n - f_{n-1})\ .$$

Using the definition $\Delta t_n := t_{n+1} - t_n$ the above expression can be tamed to look more like (A.1).

$$y_{n+1} = y_n + \Delta t_n \left[\left(1 + \frac{\Delta t_n}{2\Delta t_{n-1}}\right) f_n - \frac{\Delta t_n}{2\Delta t_{n-1}} f_{n-1}\right]$$

In the case of constant stepsize (i.e. $\Delta t_n = \Delta t_{n-1} = \Delta t$) the above relation directly reduces to the usual 2nd order Adams-Bashfourth method. As the order k increases the complexity of $p(x)$ grows rapidly. Expressions for higher orders up to 4 were computed and are shown in §A.2.3.

A.2.2 Implicit formulae

Implicit methods are constructed essentially in the same manner as the explicit methods. The key difference is the inclusion of f_{n+1} in the interpolating function. The implicit methods are of order $m + 1$. This implies that for virtually the same

memory requirements an explicit m-step predictor can use a $m+1$ order corrector. Here the interpolating function is augmented as

$$\bar{p}(x) = p(x) + \prod_{i=0}^{m-1} (x - t_{n-i}) f[t_{n+1}, t_n, \ldots, t_{n-k+1}] \,. \tag{A.5}$$

The implicit approximation for y_{n+1} is defined as

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} \bar{p}(x)\ dx \,. \tag{A.6}$$

The process of computing the coefficients is the same as before, just slightly more tedious. For an example calculation let $m = 2$. This makes the interpolating function

$$\bar{p}(x) = f_n + \frac{(x - t_{n-1})(x - t_n)}{t_{n+1} - t_{n-1}} \left(\frac{f_{n+1} - f_n}{t_{n+1} - t_n} - \frac{f_n - f_{n-1}}{t_n - t_{n-1}} \right) + \frac{(x - t_n)}{t_n - t_{n-1}} (f_n - f_{n-1}) \,.$$

Placing this in (A.6) and evaluating the integral gives

$$y_{n+1} = y_n + (t_{n+1} - t_n) f_n \frac{(t_n - t_{n+1})^2}{2(t_{n-1} - t_n)} (f_{n-1} - f_n) - \frac{(3t_{n-1} - t_n - 2t_{n+1})(t_n - t_{n+1})^2 \left(\frac{f_n - f_{n-1}}{t_{n-1} - t_n} + \frac{f_n - f_{n+1}}{t_n - t_{n+1}} \right)}{6(t_{n+1} - t_{n-1})} \,,$$

which after a little rearranging becomes

$$y_{n+1} = y_n + \frac{\Delta t_n}{6} \left[\left(2 + \frac{1}{1 + \frac{\Delta t_n}{\Delta t_{n-1}}} \right) f_{n+1} + \left(3 + \frac{\Delta t_n}{\Delta t_{n-1}} \right) f_n - \frac{1}{\frac{\Delta t_{n-1}}{\Delta t_n} \left(\frac{\Delta t_{n-1}}{\Delta t_n} + 1 \right)} f_{n-1} \right] \,.$$

Naturally, this expression also colapses to the classical Adams-Moulton form when $\Delta t_n = \Delta t_{n-1}$.

A.2.3 Compendium of variable step Adams methods to $m = 4$

The computation of the coefficients for (A.1) becomes labor intensive as m increases. Making use of the property that the coefficients are always functions of the ratio of timesteps the notation can be slightly compacted, let

$$\psi_n = \frac{\Delta t_n}{\Delta t_{n-1}}. \tag{A.7}$$

The following formulae were computed using Mathematica. The custom script also output the formalae formatted in `FortranForm` to avoid typographic errors when composing the main code.

For $m = 1$: Explicit predictor, order 1. The well known Euler Method.

$$y_{n+1} = y_n + \Delta t_n f_n \tag{A.8}$$

Implicit corrector, order 2

$$y_{n+1} = y_n + \frac{\Delta t_n}{2} \left(f_{n+1} + f_n \right) . \tag{A.9}$$

For $m = 2$: Explicit predictor, order 2

$$y_{n+1} = y_n + \Delta t_n \left(a_1 f_n + a_2 f_{n-1} \right) \tag{A.10a}$$

$$a_1 = \frac{1}{2} \left(\psi_n + 2 \right) \tag{A.10b}$$

$$a_2 = -\frac{\psi_n}{2} . \tag{A.10c}$$

Implicit corrector, order 3

$$y_{n+1} = y_n + \Delta t_n \left(a_0 f_{n+1} + a_1 f_n + a_2 f_{n-1}\right) \tag{A.11a}$$

$$a_0 = \frac{2\psi_n + 3}{6\left(\psi_n + 1\right)} \tag{A.11b}$$

$$a_1 = \frac{1}{6}\left(\psi_n + 3\right) \tag{A.11c}$$

$$a_2 = -\frac{\psi_n^2}{6\left(\psi_n + 1\right)}\,. \tag{A.11d}$$

For $m = 3$: Explicit predictor, order 3

$$y_{n+1} = y_n + \Delta t_n \left(a_1 f_n + a_2 f_{n-1} + a_3 f_{n-2}\right) \tag{A.12a}$$

$$a_1 = \frac{2\psi_{n-1}\psi_n^2 + 6\psi_{n-1}\psi_n + 3\psi_n + 6\psi_{n-1} + 6}{6\left(\psi_{n-1} + 1\right)} \tag{A.12b}$$

$$a_2 = -\frac{1}{6}\psi_n\left(2\psi_n\psi_{n-1} + 3\psi_{n-1} + 3\right) \tag{A.12c}$$

$$a_3 = \frac{\psi_{n-1}^2\psi_n\left(2\psi_n + 3\right)}{6\left(\psi_{n-1} + 1\right)}\,. \tag{A.12d}$$

Implicit corrector, order 4

$$y_{n+1} = y_n + \Delta t_n \left(a_0 f_{n+1} + a_1 f_n + a_2 f_{n-1} + a_3 f_{n-2}\right) \tag{A.13a}$$

$$a_0 = \frac{3\psi_{n-1}\psi_n^2 + 8\psi_{n-1}\psi_n + 4\psi_n + 6\psi_{n-1} + 6}{12\left(\psi_n + 1\right)\left(\psi_n\psi_{n-1} + \psi_{n-1} + 1\right)} \tag{A.13b}$$

$$a_1 = \frac{\psi_{n-1}\psi_n^2 + 4\psi_{n-1}\psi_n + 2\psi_n + 6\psi_{n-1} + 6}{12\left(\psi_{n-1} + 1\right)} \tag{A.13c}$$

$$a_2 = -\frac{\psi_n^2\left(\psi_n\psi_{n-1} + 2\psi_{n-1} + 2\right)}{12\left(\psi_n + 1\right)} \tag{A.13d}$$

$$a_3 = \frac{\psi_{n-1}^3\psi_n^2\left(\psi_n + 2\right)}{12\left(\psi_{n-1} + 1\right)\left(\psi_n\psi_{n-1} + \psi_{n-1} + 1\right)}\,. \tag{A.13e}$$

For $m = 4$: Explicit predictor, order 4

$$y_{n+1} = y_n + \Delta t_n \left(a_1 f_n + a_2 f_{n-1} + a_3 f_{n-2} + a_4 f_{n-3}\right) \tag{A.14a}$$

$$a_1 = \big[3\psi_{n-2}\psi_{n-1}^2\psi_n^3 + 12\psi_{n-2}\psi_{n-1}^2\psi_n^2 + 8\psi_{n-2}\psi_{n-1}\psi_n^2 + 4\psi_{n-1}\psi_n^2 + 18\psi_{n-2}\psi_{n-1}^2\psi_n + 6\psi_{n-2}\psi_n + 24\psi_{n-2}\psi_{n-1}\psi_n + 12\psi_{n-1}\psi_n + 6\psi_n + 12\psi_{n-2}\psi_{n-1}^2 + 12\psi_{n-2} + 24\psi_{n-2}\psi_{n-1} + 12\psi_{n-1} + 12\big] / \big[12\left(\psi_{n-1}+1\right)\left(\psi_{n-1}\psi_{n-2} + \psi_{n-2} + 1\right)\big] \tag{A.14b}$$

$$a_2 = \frac{-\psi_n}{12\left(\psi_{n-2}+1\right)}\big[3\psi_{n-2}\psi_n^2\psi_{n-1}^2 + 6\psi_{n-2}\psi_{n-1}^2 + 8\psi_{n-2}\psi_n\psi_{n-1}^2 + 12\psi_{n-2}\psi_{n-1} + 8\psi_{n-2}\psi_n\psi_{n-1} + 4\psi_n\psi_{n-1} + 6\psi_{n-1} + 6\psi_{n-2} + 6\big] \tag{A.14c}$$

$$a_3 = \frac{\psi_{n-1}^2\psi_n}{12\left(\psi_{n-1}+1\right)}\big[3\psi_{n-2}\psi_{n-1}\psi_n^2 + 4\psi_{n-2}\psi_n + 8\psi_{n-2}\psi_{n-1}\psi_n + 4\psi_n + 6\psi_{n-2} + 6\psi_{n-2}\psi_{n-1} + 6\big] \tag{A.14d}$$

$$a_4 = \frac{-\psi_{n-2}^3\psi_{n-1}^2\psi_n}{12\left(\psi_{n-2}+1\right)\left(\psi_{n-1}\psi_{n-2} + \psi_{n-2} + 1\right)}\big[3\psi_{n-1}\psi_n^2 + 8\psi_{n-1}\psi_n + 4\psi_n + 6\psi_{n-1} + 6\big]. \tag{A.14e}$$

Implicit corrector, order 5

$$y_{n+1} = y_n + \Delta t_n\left(a_0 f_{n+1} + a_1 f_n + a_2 f_{n-1} + a_3 f_{n-2} + a_4 f_{n-3}\right) \tag{A.15a}$$

$$a_0 = \big[12\psi_{n-2}\psi_{n-1}^2\psi_n^3 + 45\psi_{n-2}\psi_{n-1}^2\psi_n^2 + 30\psi_{n-2}\psi_{n-1}\psi_n^2 + 15\psi_{n-1}\psi_n^2 + 60\psi_{n-2}\psi_{n-1}^2\psi_n + 20\psi_{n-2}\psi_n + 80\psi_{n-2}\psi_{n-1}\psi_n + 40\psi_{n-1}\psi_n + 20\psi_n + 30\psi_{n-2}\psi_{n-1}^2 + 30\psi_{n-2} + 60\psi_{n-2}\psi_{n-1} + 30\psi_{n-1} + 30\big] / \big[60\left(\psi_n+1\right)\left(\psi_n\psi_{n-1} + \psi_{n-1} + 1\right)\left(\psi_{n-1}\psi_{n-2} + \psi_{n-1}\psi_n\psi_{n-2} + \psi_{n-2} + 1\right)\big] \tag{A.15b}$$

$$a_1 = \big[3\psi_{n-2}\psi_{n-1}^2\psi_n^3 + 15\psi_{n-2}\psi_{n-1}^2\psi_n^2 + 10\psi_{n-2}\psi_{n-1}\psi_n^2 + 5\psi_{n-1}\psi_n^2 + 30\psi_{n-2}\psi_{n-1}^2\psi_n + 10\psi_{n-2}\psi_n + 40\psi_{n-2}\psi_{n-1}\psi_n + 20\psi_{n-1}\psi_n + 10\psi_n + 30\psi_{n-2}\psi_{n-1}^2 + 30\psi_{n-2} + 60\psi_{n-2}\psi_{n-1} + 30\psi_{n-1} + 30\big] / \big[60\left(\psi_{n-1}+1\right)\left(\psi_{n-1}\psi_{n-2} + \psi_{n-2} + 1\right)\big] \tag{A.15c}$$

$$a_2 = -\psi_n^2\big[3\psi_{n-2}\psi_n^2\psi_{n-1}^2 + 10\psi_{n-2}\psi_{n-1}^2 + 10\psi_{n-2}\psi_n\psi_{n-1}^2 + 20\psi_{n-2}\psi_{n-1} + 10\psi_{n-2}\psi_n\psi_{n-1} + 5\psi_n\psi_{n-1} + 10\psi_{n-1} + 10\psi_{n-2} + 10\big]/\big[60\,(\psi_{n-2}+1)\,(\psi_n+1)\big] \tag{A.15d}$$

$$a_3 = \psi_{n-1}^3\psi_n^2\big[3\psi_{n-2}\psi_{n-1}\psi_n^2 + 5\psi_{n-2}\psi_n + 10\psi_{n-2}\psi_{n-1}\psi_n + 5\psi_n + 10\psi_{n-2} + 10\psi_{n-2}\psi_{n-1} + 10\big]/\big[60\,(\psi_{n-1}+1)\,(\psi_n\psi_{n-1} + \psi_{n-1} + 1)\big] \tag{A.15e}$$

$$a_4 = -\psi_{n-2}^4\psi_{n-1}^3\psi_n^2\big[3\psi_{n-1}\psi_n^2 + 10\psi_{n-1}\psi_n + 5\psi_n + 10\psi_{n-1} + 10\big]/\big[60\,(\psi_{n-2}+1)\,(\psi_{n-1}\psi_{n-2} + \psi_{n-2} + 1)\,(\psi_{n-1}\psi_{n-2} + \psi_{n-1}\psi_n\psi_{n-2} + \psi_{n-2} + 1)\big]. \tag{A.15f}$$

A.3 Implementation details of the FEM code

A.3.1 Element reference

The numbering of the finite elements used throughout this work follow from Gmsh (Geuzaine and Remacle, 2009). The common types of the elements and some of their properties are show in Table. A.2. The specific elements used in the current work are the quadratic quadrilaterals and quadratic hexahedra. The shape functions are derived in the usual manner according to the Gmsh numbering. If for example, another mesh would need to be loaded into the FEM codes then a simple reordering of the input's local node numbering to match Gmsh would be required.

The square in Fig. A.1 gives the local node numbering for the various quadrilaterals. The shape functions constructed by this numbering for the 4-node linear quadrilateral are

$$\mathcal{N}_1 = \frac{1}{4}(\eta-1)(\xi-1) \qquad \mathcal{N}_2 = \frac{1}{4}(\eta+1)(\xi-1) \tag{A.16a}$$

$$\mathcal{N}_3 = \frac{1}{4}(\eta+1)(\xi+1) \qquad \mathcal{N}_4 = \frac{1}{4}(\eta-1)(\xi+1). \tag{A.16b}$$

Table A.2: Commonly used Gmsh elements

Type	`el-type`	**No. of nodes**	**Order**	**Diagram**
Triangle	2	3	1	
	9	6	2	
Quadrilateral	3	4	1	
	10	9	2	
Tetrahedron	4	4	1	
	11	10	2	
Hexahedron	5	8	1	
	12	27	2	

For the 9-node quadratic quadrilateral the shape functions are

$$\mathcal{N}_1 = \frac{1}{4}(\eta-1)(\xi-1)\xi\eta \qquad \mathcal{N}_5 = -\frac{1}{2}(\eta-1)(\xi-1)(\xi+1)\eta \tag{A.17a}$$

$$\mathcal{N}_2 = \frac{1}{4}(\eta-1)(\xi+1)\xi\eta \qquad \mathcal{N}_6 = -\frac{1}{2}(\eta-1)(\eta+1)(\xi+1)\xi \tag{A.17b}$$

$$\mathcal{N}_3 = \frac{1}{4}(\eta+1)(\xi+1)\xi\eta \qquad \mathcal{N}_7 = -\frac{1}{2}(\eta+1)(\xi-1)(\xi+1)\eta \tag{A.17c}$$

$$\mathcal{N}_4 = \frac{1}{4}(\eta+1)(\xi-1)\xi\eta \qquad \mathcal{N}_8 = -\frac{1}{2}(\eta-1)(\eta+1)(\xi-1)\xi \tag{A.17d}$$

$$\mathcal{N}_9 = (\eta-1)(\eta+1)(\xi-1)(\xi+1). \tag{A.17e}$$

The numbering for the regular hexahedron, is shown in Fig. A.2. These elements are sometimes known as brick elements.

For a linear brick where the 8 nodes are on the vertices, as shown in Fig. A.2a,

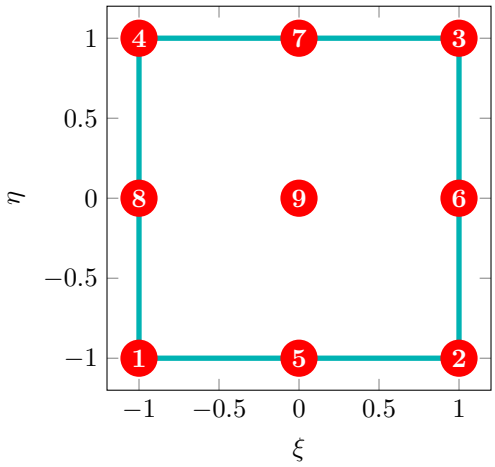
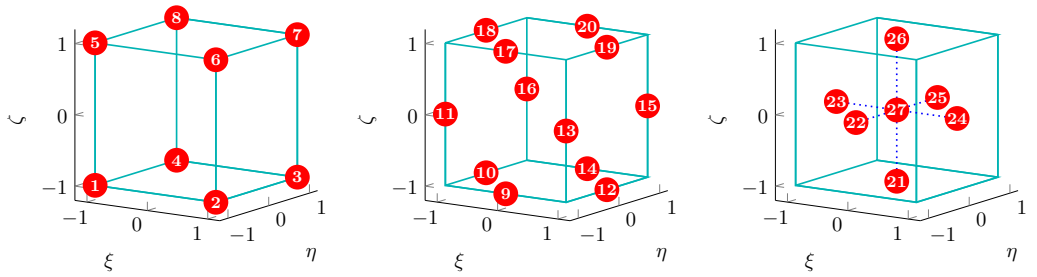


Figure A.1: Local node numbers and natural coordinates for the quadrilateral. Nodes $\{1, 2, 3, 4\}$ for are for the linear quadrilateral, and nodes 1 through 9 are for the quadratic quadrilateral.



(a) Vertex node numbers (b) Line node numbers (c) Face and center node numbers

Figure A.2: Local node numbers and natural coordinates for the hexahedron. Nodes 1 through 8 are for the linear element, and nodes 1 through 27 are for the complete quadratic hexahedron.

the shape functions are defined as the following.

$$\mathcal{N}_1 = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \qquad \mathcal{N}_2 = \frac{1}{8}(1-\xi)(\eta+1)(1-\zeta) \tag{A.18a}$$

$$\mathcal{N}_3 = \frac{1}{8}(1-\xi)(\eta+1)(\zeta+1) \qquad \mathcal{N}_4 = \frac{1}{8}(1-\xi)(1-\eta)(\zeta+1) \tag{A.18b}$$

$$\mathcal{N}_5 = \frac{1}{8}(\xi+1)(1-\eta)(1-\zeta) \qquad \mathcal{N}_6 = \frac{1}{8}(\xi+1)(\eta+1)(1-\zeta) \tag{A.18c}$$

$$\mathcal{N}_7 = \frac{1}{8}(\xi+1)(\eta+1)(\zeta+1) \qquad \mathcal{N}_8 = \frac{1}{8}(\xi+1)(1-\eta)(\zeta+1) \tag{A.18d}$$

The complete quadratic brick element has 8 nodes on the vertices, 12 on the centers of the edges, 6 on the faces, and 1 in the center. This is in contrast to various commercial implementations where the quadratic shape functions are not complete and the elements only have 20 nodes. The nodes missing are the face nodes and the center node. Using the numbering of Gmsh from Fig. A.2, the shape functions are defined as the following.

$$\mathcal{N}_1 = \frac{1}{8}(r-1)r(s-1)s(t-1)t \qquad \mathcal{N}_2 = \frac{1}{8}r(r+1)(s-1)s(t-1)t \tag{A.19a}$$

$$\mathcal{N}_3 = \frac{1}{8}r(r+1)s(s+1)(t-1)t \qquad \mathcal{N}_4 = \frac{1}{8}(r-1)rs(s+1)(t-1)t \tag{A.19b}$$

$$\mathcal{N}_5 = \frac{1}{8}(r-1)r(s-1)st(t+1) \qquad \mathcal{N}_6 = \frac{1}{8}r(r+1)(s-1)st(t+1) \tag{A.19c}$$

$$\mathcal{N}_7 = \frac{1}{8}r(r+1)s(s+1)t(t+1) \qquad \mathcal{N}_8 = \frac{1}{8}(r-1)rs(s+1)t(t+1) \tag{A.19d}$$

For the edge nodes,

$$\mathcal{N}_9 = -\frac{1}{4}\left(r^2-1\right)(s-1)s(t-1)t \quad \mathcal{N}_{10} = -\frac{1}{4}(r-1)r\left(s^2-1\right)(t-1)t \quad \text{(A.19e)}$$

$$\mathcal{N}_{11} = -\frac{1}{4}(r-1)r(s-1)s\left(t^2-1\right) \quad \mathcal{N}_{12} = -\frac{1}{4}r(r+1)\left(s^2-1\right)(t-1)t \quad \text{(A.19f)}$$

$$\mathcal{N}_{13} = -\frac{1}{4}r(r+1)(s-1)s\left(t^2-1\right) \quad \mathcal{N}_{14} = -\frac{1}{4}\left(r^2-1\right)s(s+1)(t-1)t \quad \text{(A.19g)}$$

$$\mathcal{N}_{15} = -\frac{1}{4}r(r+1)s(s+1)\left(t^2-1\right) \quad \mathcal{N}_{16} = -\frac{1}{4}(r-1)rs(s+1)\left(t^2-1\right) \quad \text{(A.19h)}$$

$$\mathcal{N}_{17} = -\frac{1}{4}\left(r^2-1\right)(s-1)st(t+1) \quad \mathcal{N}_{18} = -\frac{1}{4}(r-1)r\left(s^2-1\right)t(t+1) \quad \text{(A.19i)}$$

$$\mathcal{N}_{19} = -\frac{1}{4}r(r+1)\left(s^2-1\right)t(t+1) \quad \mathcal{N}_{20} = -\frac{1}{4}\left(r^2-1\right)s(s+1)t(t+1) \quad \text{(A.19j)}$$

finally for the face nodes and center node,

$$\mathcal{N}_{21} = \frac{1}{2}\left(r^2-1\right)\left(s^2-1\right)(t-1)t \qquad \mathcal{N}_{22} = \frac{1}{2}\left(r^2-1\right)(s-1)s\left(t^2-1\right) \quad \text{(A.19k)}$$

$$\mathcal{N}_{23} = \frac{1}{2}(r-1)r\left(s^2-1\right)\left(t^2-1\right) \qquad \mathcal{N}_{24} = \frac{1}{2}r(r+1)\left(s^2-1\right)\left(t^2-1\right) \quad \text{(A.19l)}$$

$$\mathcal{N}_{25} = \frac{1}{2}\left(r^2-1\right)s(s+1)\left(t^2-1\right) \qquad \mathcal{N}_{26} = \frac{1}{2}\left(r^2-1\right)\left(s^2-1\right)t(t+1) \quad \text{(A.19m)}$$

$$\mathcal{N}_{27} = -\left(r^2-1\right)\left(s^2-1\right)\left(t^2-1\right). \quad \text{(A.19n)}$$

A.3.2 Details of the stiffness matrix

The matrix of derivatives used in 3.13 is defined as

$$\mathbf{B} := \begin{bmatrix} \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \mathbf{N}_{,x_1}^T \mathbf{N}_{,x_1} \\ \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \mathbf{N}_{,x_2}^T \mathbf{N}_{,x_2} \\ \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \mathbf{N}_{,x_3}^T \mathbf{N}_{,x_3} \\ \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \left(\mathbf{N}_{,x_2}^T \mathbf{N}_{,x_3} + \mathbf{N}_{,x_3}^T \mathbf{N}_{,x_2}\right) \\ \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \left(\mathbf{N}_{,x_1}^T \mathbf{N}_{,x_3} + \mathbf{N}_{,x_3}^T \mathbf{N}_{,x_1}\right) \\ \left(\mathbf{x}^e+\mathbf{q}^e\right)^T \left(\mathbf{N}_{,x_1}^T \mathbf{N}_{,x_2} + \mathbf{N}_{,x_2}^T \mathbf{N}_{,x_1}\right) \end{bmatrix}. \quad \text{(A.20)}$$

Where $\mathbf{x}^e$ are the node locations of element e and $\mathbf{q}^e$ are the displacements of that element. The geometric stiffness matrix can be constructed as

$$\mathbf{K}^e_{\text{geo}} := \int_{\Omega^e_0} \mathbf{N}^T_{,x_1}\mathbf{N}_{,x_1}S_{11} + \mathbf{N}^T_{,x_2}\mathbf{N}_{,x_2}S_{22} + \mathbf{N}^T_{,x_3}\mathbf{N}_{,x_3}S_{33} + \left(\mathbf{N}^T_{,x_2}\mathbf{N}_{,x_3} + \mathbf{N}^T_{,x_3}\mathbf{N}_{,x_2}\right)S_{23} + \left(\mathbf{N}^T_{,x_1}\mathbf{N}_{,x_3} + \mathbf{N}^T_{,x_3}\mathbf{N}_{,x_1}\right)S_{13} + \left(\mathbf{N}^T_{,x_1}\mathbf{N}_{,x_2} + \mathbf{N}^T_{,x_2}\mathbf{N}_{,x_1}\right)S_{12}\ dV_0\,. \quad \text{(A.21)}$$

A.3.3 Center of mass of a deformed body in the FEM code

Computing the position of the center of mass of the deformed body can be implemented directly in the Total Lagrangian framework. The definition of center of mass of a continua is

$$\boldsymbol{r}_{COM} = \frac{\int_\Omega \boldsymbol{r}\ dm}{\int_\Omega\ dm}. \quad \text{(A.22)}$$

Where dm is an infinitesimal piece of mass located at $\boldsymbol{r}$ in the deformed body Ω. The small mass can represented by the current density and infinitesimal volume $dm = \rho\ dV$. Invoking conservation of mass the integral can be transformed from the current configuration to the reference configuration thus making the calculation simpler. Mass conservation implies that the differential mass element in the current configuration is equal to reference configuration.

$$\rho dV = \rho_0 dV_0 \quad \text{(A.23)}$$

This is used to change the integral to

$$\boldsymbol{r}_{COM} = \frac{\int_{\Omega_0} \boldsymbol{r}\rho_0\ dV_0}{\int_{\Omega_0} \rho_0\ dV_0}. \quad \text{(A.24)}$$

Now for the finite element body the centroid of element can be computed, and the body thought of as a system of particles

$$\boldsymbol{r}_{COM} = \frac{\sum_{e=1}^{n_{el}} \boldsymbol{r}_e m_e}{\sum_{e=1}^{n_{el}} m_e} \tag{A.25}$$

where m_e is the mass of the element

$$m_e = \int_{\Omega_0^e} \rho_0 \; dV_0 \tag{A.26}$$

and $\boldsymbol{r}_e$ is the location of the centroid of each continuum element

$$\boldsymbol{r}_e = \frac{1}{m_e} \int_{\Omega_0^e} \boldsymbol{r} \rho_0 \; dV_0. \tag{A.27}$$

Transforming using the usual finite element interpolating function gives a direct calculation by numerical quadrature. For an isotropic, isoparametric element this gives the mass as

$$m_e = \int_{\Omega_0^e} \rho_0 \; dV_0 = \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} \rho_0^e J \; d\xi \, d\eta \, d\zeta \tag{A.28}$$

where J is the determinant of the Jacobian of the mesh. Computing the first moment of the mass in (A.27) is performed in a similar manner using the interpolation functions $\boldsymbol{r} \rightarrow \mathbf{r} = \mathbf{N}(\xi, \eta, \zeta) \cdot (\mathbf{x} + \mathbf{q})$.

$$\int_{\Omega_0^e} \boldsymbol{r} \rho_0 \; dV_0 = \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} \mathbf{N}(\mathbf{x} + \mathbf{q}) \rho_0 J \; d\xi \, d\eta \, d\zeta \tag{A.29}$$

Therefore, to compute (A.25) a loop over the elements is used and (A.28) and (A.29) provide the information to compute the various terms. Using these expressions a consistent location for the centroid of a body undergoing arbitrary deformation can

be computed.

A.3.4 Checking the surface elements normal vector

Generating a mesh for a body begins in Gmsh by hierarchically building up points, lines, areas, and volumes. If a geometry file such as a STEP, IGES or BREP is imported from another CAD program, then basic quantities are already defined. The FEM solver implemented here relies on labeling physical entities in Gmsh as the body's volume, wet surface, and constrained surface. The surfaces in Gmsh are not necessarily oriented outward. This means that the way the code computes the normal vector

$$\boldsymbol{n} = \frac{\partial \boldsymbol{r}}{\partial \xi} \times \frac{\partial \boldsymbol{r}}{\partial \eta}$$

may be facing inward on some elements. To avoid this issue in the simulation, the offending surface elements are renumbered during pre-processing to ensure that the normal is facing outward. Since the body geometry is not degenerate, some assumptions can be made. The scheme is based on the sketch in Fig. A.3. Here the center of the volume element, node 27, is located at $\boldsymbol{a}$ and the center of the 9-node quadrilateral surface element is located at $\boldsymbol{r}$.

Let the unit vector $\hat{\boldsymbol{n}}$ be constructed as

$$\hat{\boldsymbol{n}} := \left. \frac{\frac{\partial \boldsymbol{r}}{\partial \xi} \times \frac{\partial \boldsymbol{r}}{\partial \eta}}{\left| \frac{\partial \boldsymbol{r}}{\partial \xi} \times \frac{\partial \boldsymbol{r}}{\partial \eta} \right|} \right|_{(\xi=0,\, \eta=0)} \tag{A.30}$$

where (ξ, η) are the natural coordinates of the surface element. A small motion along this vector should be located outside of the body, and away from the point $\boldsymbol{a}$ at the volume element's center. The length of $|\boldsymbol{d}|$ provides a natural scale to reference. Let

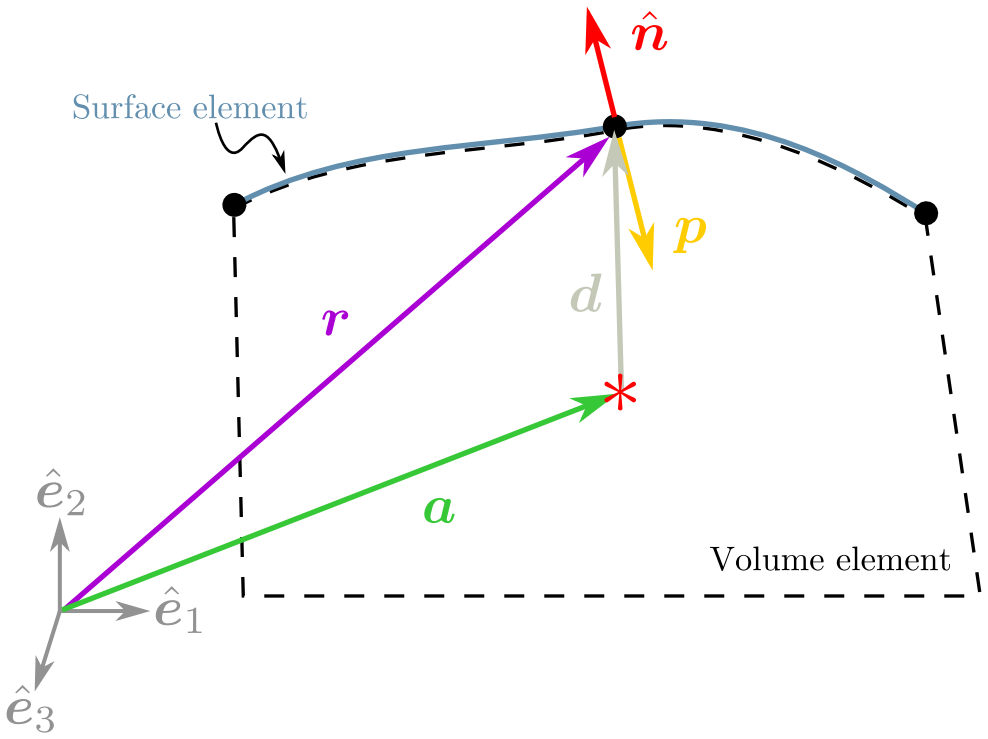


Figure A.3: Schematic of the center plane of a quadratic volume element, and the associated cross section of its surface element. This schematic defines the quantities used to ensure that $\hat{\boldsymbol{n}}$ is pointing outward.

the vector

$$\boldsymbol{p} := -\alpha|\boldsymbol{d}|\hat{\boldsymbol{n}} \tag{A.31}$$

be defined as a small motion *into* the body, where α is a small number such as $\alpha = 1/20$. If the point is inside the body then the distance from $\boldsymbol{r} + \boldsymbol{p}$ to $\boldsymbol{a}$ will be smaller than $|\boldsymbol{d}|$. If the $\hat{\boldsymbol{n}}$ was instead pointing inward, then the test would fail and we would know that the numbering of the surface elements needs to be changed. This method is summarized in Algorithm 5. The method is very efficient since it only requires a single check on each surface element relative to a known point $\boldsymbol{a}$. It does not require ray casting, winding numbers or any geometric searches over the global body. The body on the large scale can be non-convex, have holes, etc. but the method works because the checks are on the element scale. The method would fail if the angle between $\boldsymbol{d}$ and $\hat{\boldsymbol{n}}$ were greater than 90°. However this should never happen in a properly constructed FEM mesh since any complicated body will have a finely resolved mesh. Although the method is presented for a 9-node quadrilateral as the face of a 27-node hexahedron the concept would be easily generalized to other element configurations.

A.3.5 Scaling the material parameters of a body

This section describes the process of parameter selection for a FEM body. This includes the initial material density ρ_0, Young's modulus E, and Poisson's ratio ν. It is assumed that the FSI scaling is relative to some forcing frequency ω_f and a fluid density ρ_{fluid}. If the first natural frequency of the structure in the reference or initial configuration is ω_1 then the non-dimensional parameters are ν, ω_f/ω_1 and $\rho_0/\rho_{\text{fluid}}$.

Algorithm 5 Check to ensure that calculated surface normal vectors are outward facing.

Precondition: Mesh information

```
1  procedure CHECK_SURFACENORMALS
2      for e = 1 to the number of surface elements do
3          Find associated volume element
4          Extract a, r
5          Compute d = r − a
6          Compute n̂                                          ▷ (A.30)
7          Compute p                                          ▷ (A.31)
8          if |r + p − a| < |d| then
9              n̂ is outward
10         else
11             Remap local node numbers in surface connectivity array
12             {1, 2, 3, 4, 5, 6, 7, 8, 9} ↦ {2, 1, 4, 3, 5, 8, 7, 6, 9}
13         end if
14     end for
15 end procedure
```

A.3.5.1 Mass and stiffness of a homogeneous body

The density implemented in the fluid is 1, so the choice of ρ_0 selects a value of the ratio and fully determines the mass of the system. The choice of ν depends on the type of material that is being modeled. For most engineering materials, excluding rubber and other incompressibles, ν is around 0.3.

Finding the value of E is more involved. The selection and scaling of the Reynolds number usually prescribes a value for ω_f, therefore the issue becomes what value of E sets the ratio $\Omega = \omega_f/\omega_1$ as desired. In the reference configuration, let $\mathbf{M}$ and $\mathbf{K}_0$ be the mass and stiffness matrices after the essential boundary conditions are applied. The lowest eigenvalue of the algebraic eigenvalue problem

$$\mathbf{K}_0\mathbf{y} = \omega^2\mathbf{M}\mathbf{y} \tag{A.32}$$

is equal to ω_1. A simple procedure to determine E for both Kirchhoff and Biot materials begins by recognizing that for a homogeneous body, E can be factored out of the stiffness matrix. Let $\mathbf{K}_0 = E\bar{\mathbf{K}}_0$ and $\lambda = \omega^2/E$, which recasts the above eigenvalue problem as

$$\bar{\mathbf{K}}_0\mathbf{y} = \frac{\omega^2}{E}\mathbf{M}\mathbf{y} = \lambda\mathbf{M}\mathbf{y}\,.$$

Computing the first eigenvalue λ_1 of the scaled equation provides the needed relation to extract the desired E.

$$E = \frac{\omega_1^2}{\lambda_1} = \frac{\omega_f^2}{\lambda_1\Omega^2}$$

This operation is implemented using the function `eigs` in MATLAB during the preprocessing of the Gmsh produced file.

A.3.5.2 Proportional damping parameters

For simple proportional damping of the form

$$\mathbf{D} = \kappa_m\mathbf{M} + \kappa_k\mathbf{K}_0$$

where $\mathbf{M}$ is the mass matrix and $\mathbf{K}_0$ is the stiffness matrix of the reference configuration. This well-known assumed damping model has the benefit that it is diagonalized by the same eigenvectors as the undamped linear problem (Meirovitch, 2001). Computationally it also means that if stored in a compressed sparse scheme like it can use the same element pointer arrays as the other matrices in $\mathbf{K}_T$. A substantial computational cost savings is realized since a scaled addition of these matrices can be achieved with the BLAS level 1 routine `daxpy`. No special sparse BLAS is required. Generalizations of classical proportional damping exist, such as Adhikari (2006), but

the damping matrix generated will not have the same fill pattern as $\mathbf{M}$ and $\mathbf{K}$. So these methods are avoided for computational ease.

If the columns of $\mathbf{Y}$ are mass-normalized eigenvectors of (A.32), then the damping matrix becomes

$$\mathbf{Y}^T\mathbf{D}\mathbf{Y} = \kappa_m \mathbf{I} + \kappa_k \mathrm{diag}\left(\omega_1^2, \ldots, \omega_n^2\right) = 2\,\mathrm{diag}\left(\zeta_1\omega_1, \ldots, \zeta_n\omega_n\right).$$

Or the jth damping factor can be expressed as

$$\zeta_j = \frac{\kappa_m}{2\omega_j} + \frac{\kappa_k}{2}\omega_j.$$

Since we have two parameters, then we can chose only two damping factors from the entire set. The setup to find the values of (κ_m, κ_k) first revolves on what distribution of damping is desired. Picking poorly can results in unphysical values of damping at modes away from those prescribed. For example, picking the damping factor of the first two modes is likely to make the rest of damping factors strange: there will be a subset of the modes with damping factors lower than ζ_1 and ζ_2. This is caused by the mass damping term that is proportional to ω_j^{-1}. The higher modes may also have issues since they may be severely overdamped. In practice, this would make moderately high frequency disturbances persist longer than lower frequency motions, and the time step would be extremely small due to the overdamped modes. This is not a good situation.

Since the motions of interest in the FEM models here are primarily in the lower frequencies, then it is beneficial to dampen the higher modes more than the lower ones. As long as there is mass damping, there will be a dip in the values of ζ. The only way around this is to use up one of our two constants by setting $\kappa_m := 0$. So the only free parameter is κ_k, and only a single value of ζ_j can be chosen. Making sure

that none of the modes are overdamped consumes the other equation. Therefore,

$$\kappa_k = \frac{2\zeta_n}{\omega_n}.$$

The value of ω_n can be computed directly by the `eigs` function in MATLAB. In summary, if proportional damping is applied to the body, then the constants are chosen by

$$0 \leq \zeta_n \leq 1 \tag{A.33a}$$

$$\kappa_m = 0 \tag{A.33b}$$

$$\kappa_k = \frac{2\zeta_n}{\omega_n}. \tag{A.33c}$$

A.4 Continuum Mechanics

A.4.1 Isotropic Biot material

For an isotropic Biot material, the equation

$$\boldsymbol{G} = \frac{1}{2}\left(\boldsymbol{U}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{U}\right)$$

can be greatly simplified, which removes the need to compute the solution to the Lyapunov equation. First we note that principle directions of $\boldsymbol{U}$ are the principle directions of $\boldsymbol{L}$. From before, we have what appears as the eigenvalue decomposition of the tensor $\boldsymbol{U}$.

$$\boldsymbol{U} = \boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T$$

Therefore,

$$\begin{aligned}\boldsymbol{L} = \boldsymbol{U} - \boldsymbol{I} &= \boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T - \boldsymbol{I} \\ &= \boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T - \boldsymbol{\Lambda}\boldsymbol{I}\boldsymbol{\Lambda}^T \\ &= \boldsymbol{\Lambda} \underbrace{(\boldsymbol{\Sigma} - \boldsymbol{I})}_{\breve{\boldsymbol{L}}\text{: diagonal}} \boldsymbol{\Lambda}^T\end{aligned}$$

Now for an isotropic material the $\boldsymbol{L}$ is related to $\boldsymbol{G}$ from (3.19), noting the usual invariance of the trace $\mathrm{Tr}(\boldsymbol{L}) = \mathrm{Tr}(\boldsymbol{\Lambda}\breve{\boldsymbol{L}}\boldsymbol{\Lambda}^T) = \mathrm{Tr}(\breve{\boldsymbol{L}})$ we have

$$\begin{aligned}\boldsymbol{G} &= \gamma_1 \mathrm{Tr}(\boldsymbol{L})\boldsymbol{I} + \gamma_2 \boldsymbol{L} \\ &= \gamma_1 \mathrm{Tr}(\breve{\boldsymbol{L}})\boldsymbol{\Lambda}\boldsymbol{I}\boldsymbol{\Lambda}^T + \gamma_2 \boldsymbol{\Lambda}\breve{\boldsymbol{L}}\boldsymbol{\Lambda}^T \\ &= \boldsymbol{\Lambda}\left(\gamma_1 \mathrm{Tr}(\breve{\boldsymbol{L}})\boldsymbol{I} + \gamma_2 \breve{\boldsymbol{L}}\right)\boldsymbol{\Lambda}^T \\ &= \boldsymbol{\Lambda} \underbrace{\left(\gamma_1 \mathrm{Tr}(\breve{\boldsymbol{L}})\boldsymbol{I} + \gamma_2 \breve{\boldsymbol{L}}\right)}_{\breve{\boldsymbol{G}}\text{: diagonal}} \boldsymbol{\Lambda}^T \\ &= \boldsymbol{\Lambda}\breve{\boldsymbol{G}}\boldsymbol{\Lambda}^T\end{aligned}$$

So for an isotropic material, the principal directions of $\boldsymbol{U}$ are the same as the principal directions of $\boldsymbol{G}$. Placing this information back into (3.7) we have

$$\begin{aligned}\boldsymbol{G} &= \frac{1}{2}\left(\boldsymbol{U}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{U}\right) \\ \boldsymbol{\Lambda}\breve{\boldsymbol{G}}\boldsymbol{\Lambda}^T &= \frac{1}{2}\left(\boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T\boldsymbol{S} + \boldsymbol{S}\boldsymbol{\Lambda}\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T\right) \\ \breve{\boldsymbol{G}} &= \frac{1}{2}\left(\boldsymbol{\Sigma}\boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda}\boldsymbol{\Sigma}\right)\end{aligned}$$

Since the left hand side of the equation is diagonal, then the right hand side must be also. The only way to achieve this is for $\boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda}$ to be diagonal. Therefore the principal directions of $\boldsymbol{U}$, $\boldsymbol{L}$, and $\boldsymbol{G}$ are also the principal directions of $\boldsymbol{S}$. In other

words, the rotation tensor $\boldsymbol{\Lambda}$ diagonalizes all of these tensors. Now $\boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda}$ commutes with $\boldsymbol{\Sigma}$, since both are diagonal.

$$\begin{aligned}\breve{\boldsymbol{G}} &= \frac{1}{2}\left(\boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda}\right)\boldsymbol{\Sigma} \\ &= \boldsymbol{\Lambda}^T\boldsymbol{S}\boldsymbol{\Lambda}\boldsymbol{\Sigma}\end{aligned}$$

Now,

$$\begin{aligned}\boldsymbol{S} &= \boldsymbol{\Lambda}\breve{\boldsymbol{G}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Lambda}^T = \boldsymbol{\Lambda}\boldsymbol{\Sigma}^{-1}\breve{\boldsymbol{G}}\boldsymbol{\Lambda}^T \\ &= \boldsymbol{G}\boldsymbol{U}^{-1} = \boldsymbol{U}^{-1}\boldsymbol{G}.\end{aligned} \tag{A.34}$$

Using these relations gives a much simpler method to find $\boldsymbol{S}$ when compared to solving a Lyapunov equation.

A.4.2 Consistency of the Biot material with linear theory

To be consistent with classical theory a hyperelastic potential $\Psi(\lambda_1, \lambda_2, \lambda_3)$ must satisfy the following conditions (Ogden, 1984).

$$\Psi(1,1,1) = 0 \tag{A.35a}$$

$$\left.\frac{\partial\Psi}{\partial\lambda_a}\right|_{(1,1,1)} = \left.\frac{\partial\Psi}{\partial\lambda_b}\right|_{(1,1,1)} \tag{A.35b}$$

$$\left.\frac{\partial^2\Psi}{\partial\lambda_a^2}\right|_{(1,1,1)} = \left.\frac{\partial^2\Psi}{\partial\lambda_b^2}\right|_{(1,1,1)} \tag{A.35c}$$

$$\left.\frac{\partial^2\Psi}{\partial\lambda_a\partial\lambda_b}\right|_{(1,1,1)} \text{ for } a \neq b \text{ is indep. of } a \text{ and } b \tag{A.35d}$$

$$\left.\frac{\partial^2\Psi}{\partial\lambda_a^2}\right|_{(1,1,1)} - \left.\frac{\partial^2\Psi}{\partial\lambda_a\partial\lambda_b}\right|_{(1,1,1)} + \left.\frac{\partial\Psi}{\partial\lambda_a}\right|_{(1,1,1)} = \beta,\ \forall a \neq b \tag{A.35e}$$

Recall that the strain potential for the Biot model is

$$\Psi(\lambda_1,\lambda_2,\lambda_3) = \frac{1}{2}(\alpha+\beta)\left(\lambda_1^2+\lambda_2^2+\lambda_3^2\right) + \alpha\left(\lambda_1\lambda_2+\lambda_3\lambda_2+\lambda_1\lambda_3\right) - (3\alpha+\beta)\left(\lambda_1+\lambda_2+\lambda_3\right) + \left(\frac{9\alpha}{2}+\frac{3\beta}{2}\right). \quad \text{(A.36)}$$

It is now possible to take each of these preceding conditions in turn. By choice of the constant (A.35a) is satisfied. Taking the first derivative, and evaluating it at $(1,1,1)$ gives

$$\left.\frac{\partial\Psi}{\partial\lambda_a}\right|_{(1,1,1)} = 0$$

which satisfies the second condition. Taking the second derivative

$$\frac{\partial^2\Psi}{\partial\lambda_a^2} = \alpha+\beta$$

which satisfies (A.35c) for all $(\lambda_1,\lambda_2,\lambda_3)$. Computing the cross terms of the Hessian

$$\left.\frac{\partial^2\Psi}{\partial\lambda_a\partial\lambda_b}\right|_{(1,1,1)} = \alpha \text{ for } a\neq b$$

this satisfies (A.35d). Using (A.35e) the last condition is shown directly as

$$\left.\frac{\partial^2\Psi}{\partial\lambda_a^2}\right|_{(1,1,1)} - \left.\frac{\partial^2\Psi}{\partial\lambda_a\partial\lambda_b}\right|_{(1,1,1)} + \left.\frac{\partial\Psi}{\partial\lambda_a}\right|_{(1,1,1)} = (\alpha+\beta)-(\alpha)+0 = \beta$$

Therefore the isotropic Biot model is consistent with linear theory.

A.4.3 Derivatives of the invariants of stretch tensor with respect to invariants of the right Cauchy deformation tensor

To robustly compute the various derivatives of $\Psi(\bar{I}_C, \bar{I}I_C, J)$, the following fundamental identities are employed.

$$I_U^2 - 2II_U = I_C = J^{2/3}\bar{I}_C$$
$$II_U^2 - 2I_U J = II_C = J^{4/3}\bar{I}I_C$$

These identities (Hoger and Carlson, 1984) are easily verified using the definitions of the invariants in terms of the principal stretches. In ANSYS, the three first-derivatives and 6 second-derivatives are needed

$$\left\{ \frac{\partial\Psi}{\partial\bar{I}_C}, \frac{\partial\Psi}{\partial\bar{I}I_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial\bar{I}_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial\bar{I}I_C}, \frac{\partial^2\Psi}{\partial\bar{I}I_C\partial\bar{I}I_C}, \frac{\partial^2\Psi}{\partial\bar{I}_C\partial J}, \frac{\partial^2\Psi}{\partial\bar{I}I_C\partial J}, \frac{\partial\Psi}{\partial J}, \frac{\partial^2\Psi}{\partial J\partial J} \right\}.$$

Computing the first derivative of each of these expressions with respect to the scaled invariants of $\boldsymbol{C}$ yield similar forms of the equations. This can be written compactly in matrix notation.

$$\begin{bmatrix} \frac{\partial I_U}{\partial\bar{I}_C} & \frac{\partial I_U}{\partial\bar{I}I_C} & \frac{\partial I_U}{\partial J} \\ \frac{\partial II_U}{\partial\bar{I}_C} & \frac{\partial II_U}{\partial\bar{I}I_C} & \frac{\partial II_U}{\partial J} \end{bmatrix} = \begin{bmatrix} 2I_U & -2 \\ -2J & 2II_U \end{bmatrix}^{-1} \begin{bmatrix} J^{2/3} & 0 & \frac{2}{3}\bar{I}_C J^{-1/3} \\ 0 & J^{4/3} & \frac{4}{3}\bar{I}I_C J^{1/3} \end{bmatrix} \tag{A.37}$$

Likewise, the second derivatives can be constructed from knowledge of the solution of the first derivatives.

$$\begin{bmatrix} \frac{\partial^2 I_U}{\partial \bar{I}_C \partial \bar{I}_C} & \frac{\partial^2 I_U}{\partial \bar{I}_C \partial \bar{I}\bar{I}_C} & \frac{\partial^2 I_U}{\partial \bar{I}\bar{I}_C \partial \bar{I}\bar{I}_C} & \frac{\partial^2 I_U}{\partial \bar{I}_C \partial J} & \frac{\partial^2 I_U}{\partial \bar{I}\bar{I}_C \partial J} & \frac{\partial^2 I_U}{\partial J \partial J} \\ \frac{\partial^2 I\!I_U}{\partial \bar{I}_C \partial \bar{I}_C} & \frac{\partial^2 I\!I_U}{\partial \bar{I}_C \partial \bar{I}\bar{I}_C} & \frac{\partial^2 I\!I_U}{\partial \bar{I}\bar{I}_C \partial \bar{I}\bar{I}_C} & \frac{\partial^2 I\!I_U}{\partial \bar{I}_C \partial J} & \frac{\partial^2 I\!I_U}{\partial \bar{I}\bar{I}_C \partial J} & \frac{\partial^2 I\!I_U}{\partial J \partial J} \end{bmatrix} = -2 \begin{bmatrix} 2I_U & -2 \\ -2J & 2I\!I_U \end{bmatrix}^{-1}$$
$$\begin{bmatrix} \left(\frac{\partial I_U}{\partial \bar{I}_C}\right)^2 & \frac{\partial I_U}{\partial \bar{I}_C}\frac{\partial I_U}{\partial \bar{I}\bar{I}_C} & \left(\frac{\partial I_U}{\partial \bar{I}\bar{I}_C}\right)^2 & \frac{-1}{3J^{1/3}} + \frac{\partial I_U}{\partial \bar{I}_C}\frac{\partial I_U}{\partial J} \\ \left(\frac{\partial I\!I_U}{\partial \bar{I}_C}\right)^2 & \frac{\partial I\!I_U}{\partial \bar{I}_C}\frac{\partial I\!I_U}{\partial \bar{I}\bar{I}_C} & \left(\frac{\partial I\!I_U}{\partial \bar{I}\bar{I}_C}\right)^2 & -\frac{\partial I_U}{\partial \bar{I}\bar{I}_C} + \frac{\partial I\!I_U}{\partial \bar{I}\bar{I}_C}\frac{\partial I\!I_U}{\partial J} \\ & \frac{\partial I_U}{\partial \bar{I}\bar{I}_C}\frac{\partial I_U}{\partial J} & & \frac{\bar{I}_C}{9J^{4/3}} + \left(\frac{\partial I_U}{\partial J}\right)^2 \\ & -\frac{2J^{1/3}}{3} - \frac{\partial I_U}{\partial \bar{I}\bar{I}_C} + \frac{\partial I\!I_U}{\partial \bar{I}\bar{I}_C}\frac{\partial I\!I_U}{\partial J} & & -\frac{2\bar{I}\bar{I}_C}{9J^{2/3}} - 2\frac{\partial I_U}{\partial J} + \left(\frac{\partial I\!I_U}{\partial J}\right)^2 \end{bmatrix} \quad \text{(A.38)}$$

Note that these relations are nonsingular since $J > 0$ and the common term in the denominator is always greater than zero.

$$\begin{aligned} I_U I\!I_U - J &= (\lambda_1 + \lambda_2 + \lambda_3)(\lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_1\lambda_3) - \lambda_1\lambda_2\lambda_3 \\ &= (\lambda_1 + \lambda_2)(\lambda_2 + \lambda_3)(\lambda_1 + \lambda_3) > 0 \end{aligned}$$

Using equations (A.37) and (A.38) the various derivatives of Ψ can be constructed and implemented in a subroutine.

$$\begin{aligned} \frac{\partial \Psi}{\partial \bar{I}_C} &= \frac{1}{2}(\gamma_1 + \gamma_2) J^{2/3} + \gamma_1 \frac{\partial I\!I_U}{\partial \bar{I}_C} - (3\gamma_1 + \gamma_2)\frac{\partial I_U}{\partial \bar{I}_C} \\ \frac{\partial \Psi}{\partial \bar{I}_C} &= \gamma_1 \frac{\partial I\!I_U}{\partial \bar{I}\bar{I}_C} - (3\gamma_1 + \gamma_2)\frac{\partial I_U}{\partial \bar{I}\bar{I}_C} \\ &\vdots \end{aligned}$$

Unlike directly taking derivatives of (3.26), this procedure is robust, non-singular in finite arithmetic, and does not require the use of complex numbers in an implementation. For other software, such as the `UHYPER` subroutine in Abaqus, six third-derivatives are also needed. These can be constructed by extending the procedure to build the second derivatives.

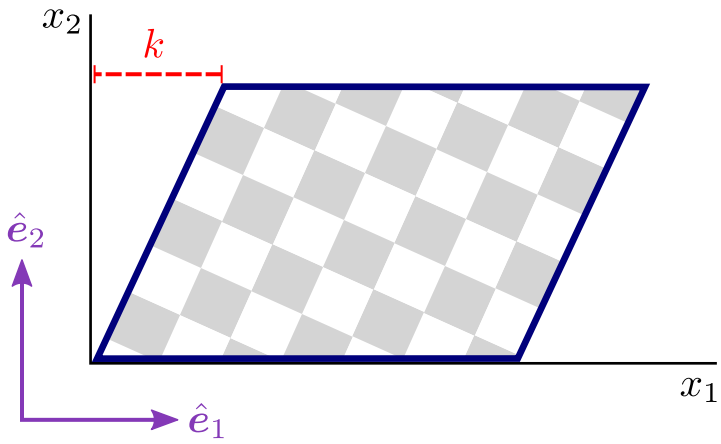


Figure A.4: Schematic of the deformation of simple shear of a rectangle.

A.4.4 Derivation of the surface tractions for simple shear

The kinematics and kinetics of the simple shearing of an isotropic elastic rectangular block are well studied (Smith, 1993). This model was also used by Farahani and Bahai (2004) to investigate a range of material models. Here this idea will be used to build the surface tractions necessary to deform a FEM model to the simple shear deformation. It is common in linear finite elements to conduct a patch test, where deformation field is known analytically for a given loading condition. The goal is to make the mesh non-uniform and still resolve the correct deformation field.

The simplest approach to building the required surface tractions is to use the known displacement field to compute the resultant stress field. Then project the surface stresses back to the reference configuration by using the definition of the Cauchy traction vector. The stress measure needed is the 1st Piola-Kirchhoff. This stress is defined in the reference configuration, and therefore it will be a constant load to force a particular amount of shear. Figure A.4 shows the schematic of the deformation field. It is important to note that the forces are applied to all the unrestrained surfaces of the block in multiple directions, and this is required to theoretically produce the linear displacement field.

The deformed configuration has the deformation gradient tensor (Smith, 1993)

$$\boldsymbol{F} = \boldsymbol{I} + k\hat{\boldsymbol{e}}_1 \otimes \hat{\boldsymbol{e}}_2 \tag{A.39}$$

throughout the entire body. In $\{\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{e}}_3\}$ components, the stretch tensor can be realized as (Farahani and Bahai, 2004)

$$\mathbf{U} = \frac{1}{\sqrt{k^2+4}} \begin{bmatrix} 2 & k & 0 \\ k & k^2+2 & \\ 0 & 0 & \sqrt{k^2+4} \end{bmatrix}. \tag{A.40}$$

For the isotropic Biot model of (3.19) the first step is to compute the symmetric Biot stress $\boldsymbol{G}$

$$\begin{aligned} \mathbf{G} &= \gamma_1 \mathrm{Tr}(\mathbf{U}-\mathbf{I})\mathbf{I} + \gamma_2(\mathbf{U}-\mathbf{I}) \\ &= \frac{1}{b} \begin{bmatrix} (b-2)(\gamma_1 b - \gamma_2) & \gamma_2 k & 0 \\ \gamma_2 k & (b-2)((\gamma_1+\gamma_2)b+\gamma_2) & 0 \\ 0 & 0 & (b-2)\gamma_1 b \end{bmatrix} \end{aligned} \tag{A.41}$$

where $b = \sqrt{k^2+4}$. Next, this stress is transformed into the 1st Piola-Kirchhoff stress $\boldsymbol{P}$ by recalling that $\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S}$ (Holzapfel, 2000).

$$\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S} = \boldsymbol{F}\left(\boldsymbol{U}^{-1}\boldsymbol{G}\right)$$

Therefore

$$\begin{aligned} \mathbf{P} &= \mathbf{F}\mathbf{U}^{-1}\mathbf{G} \\ &= \frac{1}{b} \begin{bmatrix} (2\gamma_1+\gamma_2)(b-2) & (\gamma_1(b-2)+\gamma_2(b-1))k & 0 \\ (-\gamma_1(b-2)+\gamma_2)k & (2\gamma_1+\gamma_2)(b-2) & 0 \\ 0 & 0 & \gamma_1(b-2)b \end{bmatrix} \end{aligned} \tag{A.42}$$

are the components of PK1 tensor in the Cartesian frame. The PK1 tensor is related

to the traction vector by

$$\boldsymbol{t}_0 = \boldsymbol{P}\hat{\boldsymbol{n}}_0 \tag{A.43}$$

where the subscripts are reminders that these vectors are defined on the undeformed body. As an example, the traction applied to the top surface of the body (x_2-x_3 plane) in Cartesian components are

$$\begin{aligned}
\mathbf{t}_0 &= \mathbf{P}\mathbf{e}_2 \\
&= \frac{1}{b}\begin{bmatrix} (2\gamma_1+\gamma_2)(b-2) & (\gamma_1(b-2)+\gamma_2(b-1))k & 0 \\ (-\gamma_1(b-2)+\gamma_2)k & (2\gamma_1+\gamma_2)(b-2) & 0 \\ 0 & 0 & \gamma_1(b-2)b \end{bmatrix}\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \frac{1}{b}\begin{bmatrix} (\gamma_1(b-2)+\gamma_2(b-1))k \\ (2\gamma_1+\gamma_2)(b-2) \\ 0 \end{bmatrix}
\end{aligned}$$

In a similar manner, expressions for the tractions on the other surfaces of the rectangular block can be computed.

Extending this construction, the surface stresses for virtually any material can be computed. For comparison, the surface tractions needed for a Kirchhoff material were computed by

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}\right) \tag{A.44a}$$

$$\boldsymbol{S} = \gamma_1 \mathrm{Tr}(\boldsymbol{E})\boldsymbol{I} + \gamma_2\boldsymbol{E} \tag{A.44b}$$

$$\boldsymbol{P} = \boldsymbol{F}\boldsymbol{S}. \tag{A.44c}$$

This gives PK1 in Cartesian components as

$$\mathbf{P}^{\text{Kirch.}} = \frac{1}{2}\begin{bmatrix} (\gamma_1+\gamma_2)k^2 & k\left((\gamma_1+\gamma_2)k^2+\gamma_2\right) & 0 \\ \gamma_2 k & (\gamma_1+\gamma_2)k^2 & 0 \\ 0 & 0 & \gamma_1 k^2 \end{bmatrix}. \tag{A.44d}$$

A.5 Prescribed kinematics using the Berman–Wang functions

The kinematics described by Berman and Wang (2007) are a versatile set of equations to describe a wide range of flapping motions. The function definitions are

$$\phi(t) := \frac{\phi_m}{\sin^{-1} C_\phi} \sin^{-1} \left[C_\phi \sin(\omega_f t) \right] \tag{A.45a}$$

$$\theta(t) := \theta_0 + \theta_m \cos(N\omega_f t + \Phi_\theta) \tag{A.45b}$$

$$\eta(t) := \eta_0 + \frac{\eta_m}{\tanh C_\eta} \tanh \left[C_\eta \sin(\omega_f t + \Phi_\eta) \right]. \tag{A.45c}$$

The stroke angle is specified by ϕ, the vertical deviation is θ, and the feather angle is η. Figure A.5, adapted from Berman and Wang (2007), demonstrates the novel utility of the stroke and feather functions. The stroke can be varied between a sinusoid and a triangle wave thus limiting the acceleration around the stroke reversal. The feather angle $\eta(t)$ can be set between a sinusoid and a square wave which allows for the mid stroke to have a constant prescribed angle. The vertical deviation is a regular sinusoid that, for hover, operates at twice the frequency of the other angles; $N = 2$.

These functions are adapted to parameterize the rotation matrix $\mathbf{Q}$, such that the displacements of a particular node $\mathbf{v}$ are constructed by

$$\mathbf{x} + \mathbf{v} = \mathbf{Q}\mathbf{x}. \tag{A.46}$$

Therefore

$$\mathbf{v} = (\mathbf{Q} - \mathbf{I})\mathbf{x} \tag{A.47}$$

applied to each node with a prescribed position moves the patch of the boundary as desired. The construction of the rotation matrix for a wing in the x-y plane is built

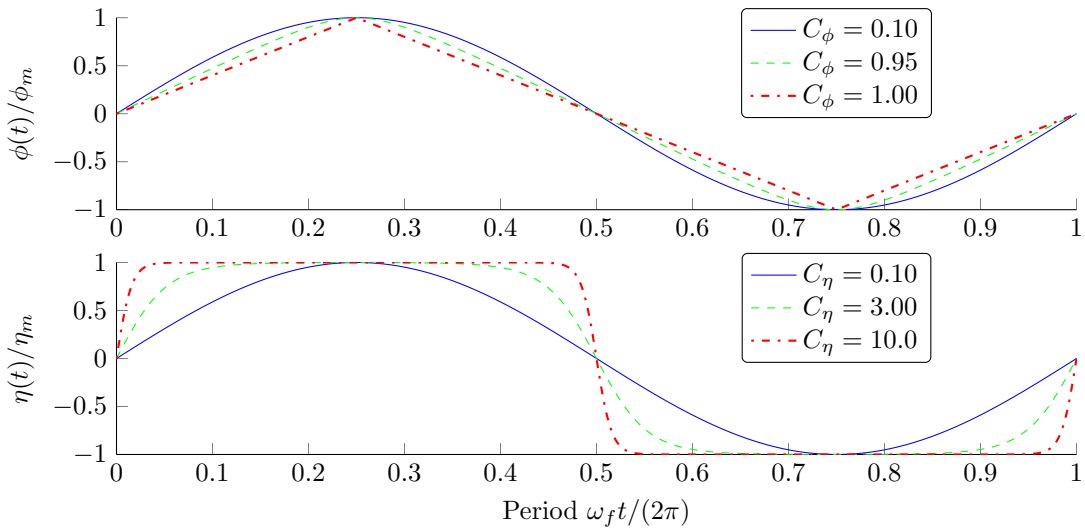


Figure A.5: Examples of the stroke angle $\phi(t)$ and the feather angle $\eta(t)$ for several values of the parameters for the Berman-Wang Kinematics.

using successive Euler rotations.

$$\begin{aligned}\mathbf{Q} &= \mathbf{Q}\left(\phi(t), \theta(t), \eta(t)\right) \\ &= \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\cos\eta & -\sin\eta \\ 0 & \sin\eta & -\cos\eta \end{bmatrix}^T \end{aligned} \tag{A.48}$$

The time derivatives $\dot{\mathbf{v}}$ and $\ddot{\mathbf{v}}$ are also required time marching. These are simply computed by using the chain rule.

$$\dot{\mathbf{v}}(t) = \dot{\mathbf{Q}}\mathbf{x} = \left(\frac{\partial \mathbf{Q}}{\partial \phi}\dot{\phi} + \frac{\partial \mathbf{Q}}{\partial \theta}\dot{\theta} + \frac{\partial \mathbf{Q}}{\partial \eta}\dot{\eta}\right)\mathbf{x} \tag{A.49}$$

$$\ddot{\mathbf{v}}(t) = \ddot{\mathbf{Q}}\mathbf{x} = \cdots \tag{A.50}$$

A visualization of these kinematics on a rigid wing are shown in Fig. A.6. The parameters are set to match a hovering hawkmoth(Berman and Wang, 2007). It is interesting to note the relatively shallow angle of the attack. The dots on the graph are equally spaced in time which provides a reference to speed of the wing tip.

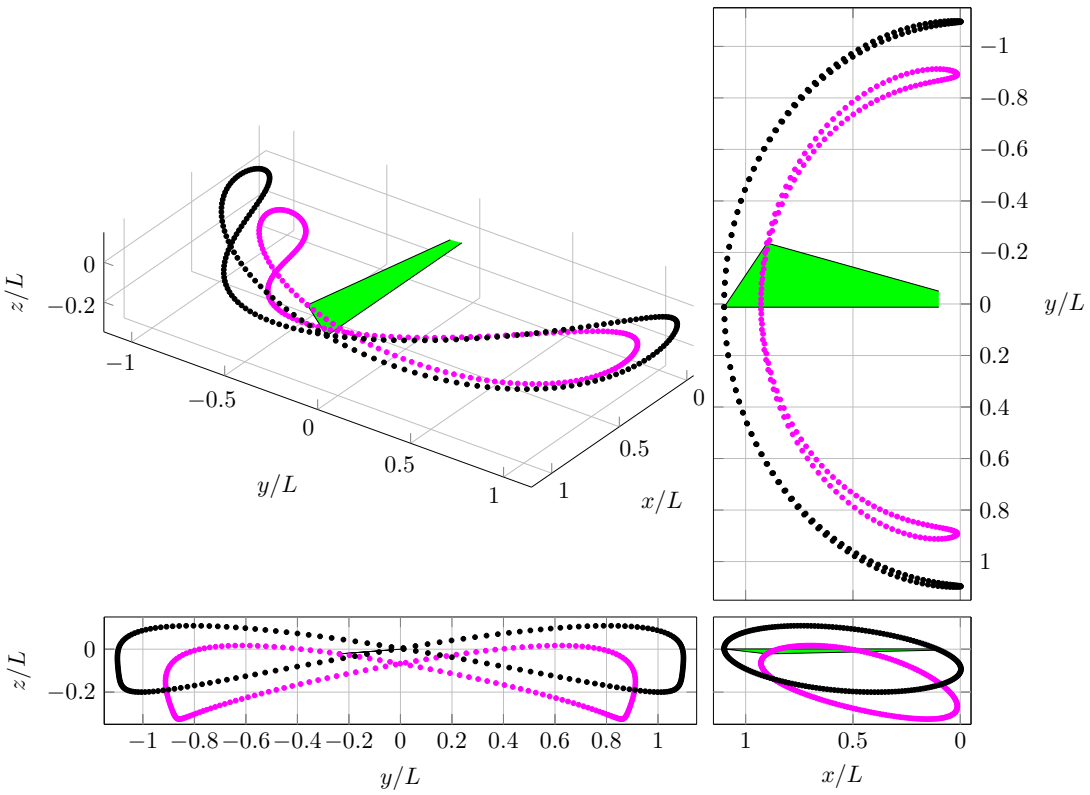


Figure A.6: Example of a rigid wing undergoing the Berman-Wang kinematics, with the parameters approximating a hovering hawkmoth. The black dots trace the wing tip, and the magenta dots trace the point of the trailing edge.

The tip dip displacement is relatively fast during the mid stroke and slow at stroke reversal. However the rotation at the end of stroke is quick.

The choice of Reynolds number involves the choice of a characteristic velocity. If this reference velocity is chosen to be the peak stroke speed then we need to compute the peak angular velocity of ϕ.

$$\max \left|\dot{\phi}(t)\right| = \frac{C_\phi \phi_m}{\sin^{-1} C_\phi} \omega_f \tag{A.51}$$

Assuming that the root of the wing is at the origin, and it is length l, then the reference velocity of the wing tip, $u^{\text{ref}} = l \max \left|\dot{\phi}(t)\right|$, can be used to compute the

forcing frequency.

$$\omega_f = \frac{u^{\text{ref}} \sin^{-1} C_\phi}{l C_\phi \phi_m} \tag{A.52}$$

Appendix B
Representative codes

Several example codes are included here as reference implementations of a Biot material. The G-α FSI, and the complete geometrically exact FEM is embedded in the `SolidMechanics` physics unit of the FLASH code. The source is available through the FLASH Center `http://flash.uchicago.edu/site/flashcode/`.

B.1 Hyperelastic user routine for ANSYS

The Fortran code below is an implementation of the `UserHyper` routines for ANSYS. This code was adapted from the template provided with the installation of ANSYS to model both isotropic, compressible Kirchhoff and Biot materials as discussed in §3.2.3.1. The inputs to the routine must be

- `nprophy`: This number is always 3, since there are 3 properties.
- The array `prophy`
  - `prophy(1)`: This selects which material model is being used. Options are 1 for Kirchhoff, or 2 for Biot.
  - `prophy(2)`: This is Young's modulus.
  - `prophy(3)`: This is Poisson's ratio.

```
!*deck,UserHyper    USERDISTRIB  parallel                                    gal
      subroutine UserHyper(                                          &
                           prophy, incomp, nprophy, invar,           &
                           potential, pInvDer)
!c***********************************************************************
!c
!c     *** user hyperelastic routine: Biot or Kirchhoff Material
!c
!c     input arguments
!c     ===============
!c      prophy       (dp,ar(*),i)       material property array
!c      nprophy      (int,sc,i)         # of material constants
!c      invar        dp,ar(3)           invariants
!c
!c     output arguments
!c     ================
!c      incomp       (log,sc,i)          fully incompressible or compressible
!c      potential     dp,sc              value of potential
!c      pInvDer       dp,ar(10)          der of potential wrt i1,i2,j
!c                                        1 - der of potential wrt i1
!c                                        2 - der of potential wrt i2
!c                                        3 - der of potential wrt i1i1
!c                                        4 - der of potential wrt i1i2
!c                                        5 - der of potential wrt i2i2
!c                                        6 - der of potential wrt i1j
!c                                        7 - der of potential wrt i2j
!c                                        8 - der of potential wrt j
!c                                        9 - der of potential wrt jj
!c
!c***********************************************************************
!c  Description:
!c               For both Kirchhoff and Biot material nprophy = 3
!c
!c               prophy(1) =1 (Kirchhoff) or =2 (Biot)
!c               prophy(2) = E, Young's Modulus
!c               prophy(3) = nu, Poisson's Ratio
!c
!c***********************************************************************
!c
```

```
!c --- parameters
!c
!c#include "impcom.inc"
      implicit none

!c
!c --- argument list
!c
      INTEGER           nprophy
      DOUBLE PRECISION  prophy(*), invar(*), potential, pInvDer(*)
      LOGICAL           incomp
!c
!c --- Select which material model to use
!c

      ! Kirchhoff Material prophy(1) = 1
      if( dabs( -prophy(1) + 1.d0 ) <= 1.d-3 ) then
            call UserHyper_Kirch(prophy, incomp, nprophy, invar,   &
                                 potential, pInvDer)

      ! Kirchhoff Material prophy(1) = 2
      else if( dabs( -prophy(1) + 2.d0 ) <= 1.d-3 ) then
            call UserHyper_Biot( prophy, incomp, nprophy, invar,   &
                                 potential, pInvDer)

      else
         print*,'Error: '

      end if

      RETURN

      END subroutine UserHyper

!c*** user hyperelastic routine: Kirchhoff Material
      subroutine UserHyper_Kirch(                                  &
                           prophy, incomp, nprophy, invar,         &
                           potential, pInvDer)
      implicit none

!c
!c --- argument list
!c
      INTEGER           nprophy
      DOUBLE PRECISION  prophy(*), invar(*), potential, pInvDer(*)
      LOGICAL           incomp
!c
!c --- local variables
!c
      DOUBLE PRECISION Ib, IIb, J, alpha, beta, E, nu, J13, J23, J43

      ! Initialize the local variables
      Ib  = invar(1)
      IIB = invar(2)
      J   = invar(3)
      ! prophy(1) : material model
      E   = prophy(2)
      nu  = prophy(3)

      J13 = J**(1.d0/3.d0)
      J23 = J**(2.d0/3.d0)
      J43 = J**(4.d0/3.d0)

      ! intialize the outputs
      incomp = .false.
      !potential = 0.d0
      !pInvDer(1:9) = 0.d0

      ! set the material properties (alpha,beta)
      alpha = E*nu/(1.d0 + nu)/(1.d0-nu-nu)
      beta = E/(1.d0 + nu)

      potential = ((-3 + Ib*J23)**2*alpha +(3 - 2*Ib*J23 + (Ib**2 - 2*IIb)*J43)*beta)/8.d0

      pInvDer(1:9) = (/                                                                    &
                (J23*((-3 + Ib*J23)*alpha + (-1 + Ib*J23)*beta))/4.,                       &
                -(J43*beta)/4.,                                                            &
                (J43*(alpha + beta))/4.,                                                   &
                0.d0,                                                                      &
                0.d0,                                                                      &
                ((-3 + 2*Ib*J23)*alpha + (-1 + 2*Ib*J23)*beta)/(6*J13),                    &
                -(J13*beta)/3.,                                                            &
                (-2*IIb*J23*beta + Ib**2*J23*(alpha + beta) - Ib*(3*alpha + beta))/(6*J13), &
                (-2*IIb*J23*beta + Ib**2*J23*(alpha + beta) + Ib*(3*alpha + beta))/(18*J43) &
                /)

      RETURN

      END subroutine UserHyper_Kirch

!c*** user hyperelastic routine: Biot Material
      subroutine UserHyper_Biot(                                   &
```

```
                              prophy, incomp, nprophy, invar,     &
                              potential, pInvDer)
      implicit none

!c
!c --- argument list
!c
      INTEGER           nprophy
      DOUBLE PRECISION  prophy(*), invar(*), potential, pInvDer(*)
      LOGICAL           incomp
!c
!c --- local variables
!c
      DOUBLE PRECISION  Ib, IIb,i1,i2, J, alpha, beta, E, nu, &
                        A1, A2, J13, J23, J43, J53, J73, J83, &
                        Di1(9), Di2(9)

      ! Initialize the local variables
      Ib  = invar(1)
      IIb  = invar(2)
      J   = invar(3)

      ! prophy(1) : material model
      E   = prophy(2)
      nu  = prophy(3)

      J13 = J**(1.d0/3.d0)
      J23 = J**(2.d0/3.d0)
      J43 = J**(4.d0/3.d0)
      J53 = J**(5.d0/3.d0)
      J73 = J**(7.d0/3.d0)
      J83 = J**(8.d0/3.d0)

      ! intialize the outputs
      incomp       = .false.
      potential    = 0.d0
      pInvDer(1:9) = 0.d0

      ! set the material properties (alpha,beta)
      alpha = E*nu/(1.d0 + nu)/(1.d0-nu-nu)
      beta  = E/(1.d0 + nu)
      A1    = 0.5d0*(alpha+beta)
      A2    = (3*alpha + beta)

      ! Compute the Invariants of U
      i1 = J13*phi(Ib,IIb)
      i2 = J23*phi(IIb,Ib)

      ! Compute the potential
      potential = A1*J23*Ib + alpha*i2 - A2*i1 + 1.5d0*(3*alpha + beta)

      ! Compute the derivatives of i1
      Di1 = (/                                                          &
          (i2*J23)/(2*i1*i2 - 2*J),                                     &
          J43/(2*i1*i2 - 2*J),                                          &
          (J43*(i2**3 + J**2))/(4.*(-(i1*i2) + J)**3),                  &
          (J**2*(i2**2 + i1*J))/(4.*(-(i1*i2) + J)**3),                 &
          ((i1**2 + i2)*J83)/(4.*(-(i1*i2) + J)**3),                    &
          (i2**3*(-2*i1**2 + Ib*J23) + 2*i2**2*(2*i1 + IIb*J13)*J       &
                + 3*i1**2*J**2 + i2*J**2 + 2*i1*IIb*J73                 &
                + Ib*J83)/(6.*J13*(-(i1*i2) + J)**3),                   &
          (-4*i1**2*i2**2*J13 + i2**2*Ib*J + i1*(3*i1**2                &
                + 8*i2)*J43 + 2*(i1**2 + i2)*IIb*J53 + i1*Ib*J**2       &
                - J73)/(6.*(-(i1*i2) + J)**3),                          &
          (3*i1 + (i2*Ib)/J13 + 2*IIb*J13)/(3*i1*i2 - 3*J),             &
          -((2*IIb)/J23 - (3*i1**2 + 2*i1*IIb*J13 + Ib*J23)**2          &
                /(-(i1*i2) + J)**2 - (6*(3*i1 + (i2*Ib)/J13             &
                + 2*IIb*J13))/(-(i1*i2) + J) + i2*(-(Ib/J43)            &
                - (3*i1 + (i2*Ib)/J13 + 2*IIb*J13)**2/(-(i1*i2)         &
                + J)**2))/(9.*(-(i1*i2) + J))                           &
          /)

      ! Compute the derivatives of i2
      Di2 = (/                                                          &
          J53/(2*i1*i2 - 2*J),                                          &
          (i1*J43)/(2*i1*i2 - 2*J),                                     &
          (J73*(i2**2 + i1*J))/(4.*(-(i1*i2) + J)**3),                  &
          ((i1**2 + i2)*J**3)/(4.*(-(i1*i2) + J)**3),                   &
          (J83*(i1**3 + J))/(4.*(-(i1*i2) + J)**3),                     &
          (J23*(-5*i1**2*i2**2 + i2**2*Ib*J23 +    i1*(3*i1**2          &
                + 10*i2)*J + 2*(i1**2 + i2)*IIb*J43 + i1*Ib*J53         &
                - 2*J**2))/(6.*(-(i1*i2) + J)**3),                      &
          (J13*(-4*i1**3*i2**2 + i1**2*(3*i1**2 + 5*i2)*J +             &
                2*i1**3*IIb*J43 + (i1**2 + i2)*Ib*J53 + 2*i1*J**2       &
                + 2*IIb*J73))/(6.*(-(i1*i2) + J)**3),                   &
          (3*i1**2 + 2*i1*IIb*J13 + Ib*J23)/(3*i1*i2 - 3*J),            &
          (Ib/J13 + ((i2*Ib + 3*i1*J13 + 2*IIb*J23)**2*J13)/            &
                (-(i1*i2) + J)**2 + i1*((-2*IIb)/J23 + (3*i1**2         &
                + 2*i1*IIb*J13 + Ib*J23)**2/(-(i1*i2) + J)**2           &
                + (6*(3*i1 + (i2*Ib)/J13 + 2*IIb*J13))/(-(i1*i2)        &
                + J)))/(9.*(-(i1*i2) + J))                              &
          /)
```

```
224
225
226       !(1): D[pot,Ib]  der of potential wrt Ib
227       pInvDer(1) = A1*J23 + alpha*Di2(1) - A2*Di1(1)
228
229       !(2): D[pot,IIb]  der of potential wrt IIb
230       pInvDer(2) = alpha*Di2(2) - A2*Di1(2)
231
232       !(3): D[pot,Ib,Ib]  der of potential wrt i1i1
233       pInvDer(3) = alpha*Di2(3) - A2*Di1(3)
234
235       !(4): D[pot,Ib,IIb]  der of potential wrt i1i2
236       pInvDer(4) = alpha*Di2(4) - A2*Di1(4)
237
238       !(5): D[pot,IIb,IIb]  der of potential wrt i2i2
239       pInvDer(5) = alpha*Di2(5) - A2*Di1(5)
240
241       !(6): D[pot,Ib,J]  der of potential wrt i1J
242       pInvDer(6) = 2*A1/(3*J13) + alpha*Di2(6) - A2*Di1(6)
243
244       !(7): D[pot,IIb,J]  der of potential wrt i2J
245       pInvDer(7) = alpha*Di2(7) - A2*Di1(7)
246
247       !(8): D[pot,J]  der of potential wrt J
248       pInvDer(8) = 2*A1*Ib/(3*J13) + alpha*Di2(8) - A2*Di1(8)
249
250       !(9): D[pot,J,J]  der of potential wrt JJ
251       pInvDer(9) = -2*A1*Ib/(9*J43) + alpha*Di2(9) - A2*Di1(9)
252
253       RETURN
254
255       CONTAINS
256
257         function zeta(x,y)
258             implicit none
259             double precision :: zeta
260             double precision, intent(in) :: x,y
261
262             if( dabs( x**2 - 3*y ) <= 1.d-12 ) then
263                 zeta = 0.d0
264             else
265                 zeta = (27 + 2*x**3 - 9*x*y)/(2*(x**2 - 3*y)**1.5d0 )
266             end if
267         end function zeta
268
269         function g(x,y)
270             implicit none
271             double precision :: g
272             double precision, intent(in) :: x,y
273             complex*16 :: z, z2, xc, yc, temp
274             xc = dcmplx(x,0.d0)
275             yc = dcmplx(y,0.d0)
276             z  = dcmplx(zeta(x,y), 0.d0)
277             z2 = cdsqrt( z*z - 1.d0 )
278             temp = cdSqrt(xc + cdSqrt(xc**2 - 3*yc)*((z - z2)**(1.d0/3.d0) &
279                    + (z + z2)**(1.d0/3.d0)))/dSqrt(3.d0)
280             g = REAL(temp)
281         end function
282
283         function phi(x,y)
284             implicit none
285             double precision :: phi
286             double precision, intent(in) :: x,y
287             double precision :: gg
288             gg = g(x,y)
289             phi = gg + dSqrt(2.d0/gg - gg**2 + x)
290         end function
291
292       END subroutine UserHyper_Biot
```

B.2 Example element stiffness matrix for the Biot material in MATLAB

The code below is written in MATLAB as a prototype of the element stiffness calculations implemented in Fortran for FLASH. It computes the internal force vector `Qs` and stiffness matrix `ke` for a 27-node hexahedron element. The inputs are

- `X`: The reference nodal locations of the element.
- `qe`: The DOF of the element.

- `E`: Young's Modulus.
- `nu`: Poisson's Ratio.
- `quad_nt`: The number of quadrature points.
- `quad_xi`, `quad_eta`, `quad_zeta`: Lists of the locations of the quadrature points in the natural coordinates (ξ, η, ζ) of the isoparametric element.
- `quad_w`: The list of quadrature weights.

```
%% Biot-material, 27-node hexahedron
function [ke Qs] = el12_ke_biot(X, qe, E, nu, ...
                    quad_nt, quad_xi, quad_eta, quad_zeta, quad_w) %#codegen

%% Define some sizing constants
ned = 3;
nen_e = 27;
nee = 81;

%% initialize the matrices
kb = zeros(nee,nee);
ks = zeros(nee,nee);
Qs = zeros(nee,1);

%% Define Integration loop
for i1 = 1:quad_nt
    xi = quad_xi(i1);
    eta = quad_eta(i1);
    zeta = quad_zeta(i1);

    %% Get Shape Functions, and local derivatives
    [~,NNxi,NNeta,NNzeta] = el12_ShapeFunctions(xi,eta,zeta);

    %% Build [N]
    %N     = expand_shapeNN(NN    , nen_e, ned);
    Nxi   = expand_shapeNN(NNxi  , nen_e, ned);
    Neta  = expand_shapeNN(NNeta , nen_e, ned);
    Nzeta = expand_shapeNN(NNzeta, nen_e, ned);

    %% Compute Jacobian
    J = [Nxi*X, Neta*X, Nzeta*X];
    detJ = det(J);

    %% Derivative matrix DN
    DN = [NNxi', NNeta', NNzeta']/J;

    %% Build Nx, Ny, Nz
    Nx = expand_shapeNN(DN(:,1), nen_e, ned);
    Ny = expand_shapeNN(DN(:,2), nen_e, ned);
    Nz = expand_shapeNN(DN(:,3), nen_e, ned);

    %% Build DXX, DYY, DZZ, DYZ, DXZ, DXY
    DXX = Nx'*Nx;
    DYY = Ny'*Ny;
    DZZ = Nz'*Nz;
    DYZ = Ny'*Nz;
    DXZ = Nx'*Nz;
    DXY = Nx'*Ny;

    %% Build B
    B = [ (X + qe)'*DXX;
         (X + qe)'*DYY;
         (X + qe)'*DZZ;
         (X + qe)'*(DYZ' + DYZ);
         (X + qe)'*(DXZ' + DXZ);
         (X + qe)'*(DXY' + DXY)];

    %% Get the stress: {S} and [DS]
    F = [ Nx*qe, Ny*qe, Nz*qe ] + eye(3);
    [Psi, Sigma, Lambda] = svd(F);
    U = Lambda*Sigma*Lambda';
    R = Psi*Lambda';
    Tr_U = Sigma(1,1)+Sigma(2,2)+Sigma(3,3);
    Y = Tr_U*eye(3)-U;
    detY = det(Y);
    L = U-eye(3);
    lambda = E*nu/(1+nu)/(1-2*nu);
    mu = E/2/(1+nu);
    Tr_L = L(1,1)+L(2,2)+L(3,3);
    G = lambda*Tr_L*eye(3) + 2*mu*L;
    S = U\G;
    DS = zeros(6,nee);
```

```
    for k = 1:nee
        dF = [Nx(:,k),Ny(:,k),Nz(:,k)];
        theta = R'*dF-1/detY*Y*(R'*dF-dF'*R)*Y*U;
        Tr_theta = theta(1,1)+theta(2,2)+theta(3,3);
        dS = (lambda*Tr_theta*eye(3)+2*mu*theta-S*theta)/U;
        DS(:,k) = [dS(1,1);
            dS(2,2);
            dS(3,3);
            dS(2,3);
            dS(1,3);
            dS(1,2)];
    end
    S = [S(1,1);
        S(2,2);
        S(3,3);
        S(2,3);
        S(1,3);
        S(1,2)];

    %% compute dV0
    dV0 = detJ*quad_w(i1);

    %% Build kb
    kb = kb + B'*DS*dV0;

    %% Build ks
    ks = ks + (DXX*S(1) + DYY*S(2) + DZZ*S(3) ...
        + (DYZ'+DYZ)*S(4) + (DXZ'+DXZ)*S(5) + (DXY'+DXY)*S(6))*dV0;

    %% Build Qs
    Qs = Qs + B'*S*dV0;
end
%% Output element stiffness:
ke = kb+ks;
```

Bibliography

ABAQUS (2012). *Abaqus User Subroutines Reference Manual*. Dassault Systémes.

Adhikari, S. (2006). Damping modelling using generalized proportional damping. *Journal of Sound and Vibration* **293**(1-2), 156–170. doi:10.1016/j.jsv.2005.09.034.

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., and Sorensen, D. (1999). *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 3rd edition. doi:10.1137/1.9780898719604.

ANSYS (2009). *Programmer's Manual for Mechanical APDL*. ANSYS, Inc.

Antman, S.S. (2004). *Nonlinear Problems of Elasticity*, volume 107 of *Applied Mathematical Sciences*. Springer. doi:10.1007/0-387-27649-1.

ASC Flash Center (2012). Flash user's guide. `http://flash.uchicago.edu/site/flashcode/user_support/flash4_ug.pdf`.

Balay, S., Brown, J., , Buschelman, K., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., McInnes, L.C., Smith, B.F., and Zhang, H. (2013). PETSc users manual. Technical Report ANL-95/11 - Revision 3.4, Argonne National Laboratory. `http://www.mcs.anl.gov/petsc`.

Baldo, G., Bonelli, A., Bursi, O., and Erlicher, S. (2006). The accuracy of the generalized-α method in the time integration of non-linear single- and two-DOF forced systems. *Computational Mechanics* **38**, 15–31. doi:10.1007/s00466-005-0718-x.

Bauchau, O.A. (2011). *Flexible Multibody Dynamics*, volume 176 of *Solid Mechanics and its Applications*. Springer. doi:10.1007/978-94-007-0335-3.

Belytschko, T., Liu, W.K., and Moran, B. (2000). *Nonlinear finite elements for continua and structures*. Wiley.

Berman, G.J. and Wang, Z.J. (2007). Energy-minimizing kinematics in hovering insect flight. *Journal of Fluid Mechanics* **582**, 153–168. doi:10.1017/S0022112007006209.

Bernard, P.S. (2011). The hairpin vortex illusion. *Journal of Physics: Conference Series* **318**(6), 062004. doi:10.1088/1742-6596/318/6/062004.

Bonelli, A., Bursi, O.S., Erlicher, S., and Vulcan, L. (2002). Analyses of the generalized-α method for linear and non-linear forced excited systems. In *Structural Dynamics-EURODYN 2002*, volume 2, 1523–1528. Munich, Germany.

Bouby, C., Fortun, D., Pietraszkiewicz, W., and Valle, C. (2005). Direct determination of the rotation in the polar decomposition of the deformation gradient by maximizing a rayleigh quotient. *Zeitschrift fr Angewandte Mathematik und Mechanik* **85**(3), 155–162. doi:10.1002/zamm.200310167.

Carroll, M.M. (2004). Derivatives of the rotation and stretch tensors. *Mathematics and Mechanics of Solids* **9**(5), 543–553. doi:10.1177/1081286504038674.

Chen, Y. and Wheeler, L. (1993). Derivatives of the stretch and rotation tensors. *Journal of Elasticity* **32**(3), 175–182. doi:10.1007/BF00131659.

Chung, J. and Hulbert, G.M. (1993). A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized-α method. *Journal of Applied Mechanics* **60**, 371–375. doi:10.1115/1.2900803.

Cleaver, D., Calderon, D.E., Wang, Z.J., and Gursul, I. (2013a). Low aspect ratio oscillating flexible wings at low reynolds numbers. In *43rd Fluid Dynamics Conference*. AIAA, San Diego, CA. doi:10.2514/6.2013-3178.

Cleaver, D., Wang, Z.J., and Gursul, I. (2013b). Oscillating flexible wings at low reynolds numbers. In *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. AIAA, Grapevine (Dallas/Ft. Worth Region), Texas. doi:10.2514/6.2013-674.

Combes, S.A. and Daniel, T.L. (2003a). Flexural stiffness in insect wings I. scaling and the influence of wing venation. *Journal of Experimental Biology* **206**, 2979–2987. doi:10.1242/jeb.00523.

Combes, S.A. and Daniel, T.L. (2003b). Flexural stiffness in insect wings II. spatial distribution and dynamic wing bending. *Journal of Experimental Biology* **206**, 2979–2987. doi:10.1242/jeb.00524.

Combes, S.A. and Daniel, T.L. (2003c). Into thin air: contributions of aerodynamic and inertial-elastic forces to wing bending in the hawkmoth manduca sexta. *Journal of Experimental Biology* **206**, 2999–3006. doi:10.1242/jeb.00502.

Comstock, J.H. (1918). *The wings of insects*. Comstock Publishing Company, Ithaca, New York.

Dai, H., Luo, H., and Doyle, J.F. (2012). Dynamic pitching of an elastic rectangular wing in hovering motion. *Journal of Fluid Mechanics* **693**, 473–499. doi:10.1017/jfm.2011.543.

Daley, C., Vanella, M., Dubey, A., Weide, K., and Balaras, E. (2012). Optimization of multigrid based elliptic solver for large scale simulations in the FLASH code. *Concurrency and Computation: Practice and Experience* **24**(18), 2346–2361. doi:10.1002/cpe.2821.

Daniel, T.L. and Combes, S.A. (2002). Flexible wings and fins: bending by inertial or fluid dynamics forces? *Integrative Comparative Biology* **42**(5), 1044–1049. doi:10.1093/icb/42.5.1044.

Davis, T.A. (2004). Algorithm 832: UMFPACK V4.3–an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software* **30**(2), 196–199. doi:10.1145/992200.992206.

Degroote, J. (2013). Partitioned simulation of fluid-structure interaction. *Archives of Computational Methods in Engineering* **20**(3), 185–238. doi:10.1007/s11831-013-9085-5.

Demmel, J.W., Eisenstat, S.C., Gilbert, J.R., Li, X.S., and Liu, J.W.H. (1999). A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Analysis and Applications* **20**(3), 720–755. doi:10.1137/S0895479895291765.

Deng, X., Schenato, L., Wu, W.C., and Sastry, S. (2006a). Flapping flight for biomimetic robotic insects: Part I-system modeling. *IEEE Transactions on Robotics* **22**(4), 776–788. doi:10.1109/TRO.2006.875480.

Deng, X., Schenato, L., Wu, W.C., and Sastry, S. (2006b). Flapping flight for biomimetic robotic insects: Part II-flight control design. *IEEE Transactions on Robotics* **22**(4), 789–803. doi:10.1109/TRO.2006.875483.

Dhooge, A., Govaerts, W., and Kuznetsov, Y.A. (2003). MATCONT: A MATLAB package for numerical bifurcation analysis of ODEs. *ACM Transactions on Mathematical Software* **29**(2), 141–164. doi:10.1145/779359.779362.

Dhooge, A., Govaerts, W., Kuznetsov, Y.A., Meijer, H.G., and Sautois, B. (2008). New features of the software MatCont for bifurcation analysis of dynamical systems. *Mathematical and Computer Modelling of Dynamical Systems* **14**(2), 147–175. doi:10.1080/13873950701742754.

Dickinson, M.H., Lehmann, F.O., and Sane, S.P. (1999). Wing rotation and the aerodynamic basis of insect flight. *Science* **284**(5422), 1954–1960. doi:10.1126/science.284.5422.1954.

Dirks, J.H. and Taylor, D. (2012a). Fracture toughness of locust cuticle. *The Journal of Experimental Biology* **215**(9), 1502–1508. doi:10.1242/jeb.068221.

Dirks, J.H. and Taylor, D. (2012b). Veins improve fracture toughness of insect wings. *PLoS ONE* **7**(8), e43411. doi:10.1371/journal.pone.0043411.

Doman, D.B., Oppenheimer, M.W., and Sigthorsson, D.O. (2010). Wingbeat shape modulation for flapping-wing micro-air-vehicle control during hover. *Journal of Guidance, Control, and Dynamics* **33**(3), 724–739. doi:10.2514/1.47146.

Ellington, C.P. (1984a). The aerodynamics of hovering insect flight. I. the quasi-steady analysis. *Philosophical Transactions of the Royal Society of London* **305**(1122), 1–15. doi:10.1098/rstb.1984.0049.

Ellington, C.P. (1984b). The aerodynamics of hovering insect flight. II. morphological parameters. *Philosophical Transactions of the Royal Society of London* **305**(1122), 17–40. doi:10.1098/rstb.1984.0050.

Ellington, C.P. (1984c). The aerodynamics of hovering insect flight. III. kinematics. *Philosophical Transactions of the Royal Society of London* **305**(1122), 41–78. doi:10.1098/rstb.1984.0051.

Ellington, C.P. (1984d). The aerodynamics of hovering insect flight. IV. aerodynamic mechanisms. *Philosophical Transactions of the Royal Society of London* **305**(1122), 79–113. doi:10.1098/rstb.1984.0052.

Ellington, C.P. (1984e). The aerodynamics of hovering insect flight. V. a vortex theory. *Philosophical Transactions of the Royal Society of London* **305**(1122), 115–144. doi:10.1098/rstb.1984.0053.

Ellington, C.P. (1984f). The aerodynamics of hovering insect flight. VI. lift and power requirements. *Philosophical Transactions of the Royal Society of London* **305**(1122), 145–181. doi:10.1098/rstb.1984.0054.

Ellington, C.P., van den Berg, C., Willmott, A.P., and Thomas, A.L.R. (1996). Leading-edge vortices in insect flight. *Nature* **384**, 626–630. doi:10.1038/384626a0.

Ennos, A.R. (1989). Inertial and aerodynamic torques on the wings of Diptera in flight. *Journal of Experimental Biology* **142**(1), 87–95.

Erlicher, S., Bonaventura, L., and Bursi, O.S. (2002). The analysis of the generalized-α method for non-linear dynamic problems. *Computational Mechanics* **28**(2), 83–104. doi:10.1007/s00466-001-0273-z.

Erzincanli, B. and Sahin, M. (2013). An arbitrary lagrangianeulerian formulation for solving moving boundary problems with large displacements and rotations. *Journal of Computational Physics* **255**, 660–679. doi:10.1016/j.jcp.2013.08.038.

Farahani, K. and Bahai, H. (2004). Hyper-elastic constitutive equations of conjugate stresses and strain tensors for the Seth–Hill strain measures. *International Journal of Engineering Science* **42**(1), 29–41. doi:10.1016/S0020-7225(03)00241-6.

Farahani, K. and Naghdabadi, R. (2003). Basis free relations for the conjugate stresses of the strains based on the right stretch tensor. *International Journal of Solids and Structures* **40**(22), 5887–5900. doi:10.1016/S0020-7683(03)00294-4.

Farmer, J.D., Ott, E., and Yorke, J.A. (1983). The dimension of chaotic attractors. *Physica D* **7**(1-3), 153–180. doi:10.1016/0167-2789(83)90125-2.

Felippa, C.A. and Haugen, B. (2005). A unified formulation of small-strain corotational finite elements:I. Theory. *Computer Methods in Applied Mechanics and Engineering* **194**(21-24), 2285–2335. doi:10.1016/j.cma.2004.07.035.

Felippa, C.A., Park, K.C., and Farhat, C. (2001). Partitioned analysis of coupled mechanical systems. *Computer Methods in Applied Mechanics and Engineering* **190**(2425), 3247–3270. doi:10.1016/S0045-7825(00)00391-1.

Fitzgerald, T., Valdez, M., Vanella, M., Balaras, E., and Balachandran, B. (2011). Flexible flapping systems: Computational investigations into fluid-structure interactions. *The Aeronautical Journal* **115**(1172).

Fry, S.N., Sayaman, R., and Dickinson, M.H. (2005). The aerodynamics of hovering flight in Drosophila. *Journal of Experimental Biology* **208**, 2303–2318. doi:10.1242/jeb.01612.

Geuzaine, C. and Remacle, J.F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* **79**(11), 1309–1331. doi:10.1002/nme.2579.

Hairer, E., Nørsett, S.P., and Wanner, G. (1993). *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer.

HDF Group (2013). Hierarchial data format version 5. `http://www.hdfgroup.org/HDF5`.

Heathcote, S. and Gursul, I. (2007). Flexible flapping airfoil propulsion at low reynolds numbers. *AIAA Journal* **45**(5), 1066–1079. doi:10.2514/1.25431.

Hedrick, T.L. and Daniel, T.L. (2006). Flight control in the hawkmoth Manduca sexta: the inverse problem of hovering. *Journal of Experimental Biology* **209**, 3114–3130. doi:10.1242/jeb.02363.

Herbert, R.C. (2001). *Modelling insect wings using the finite element method*. Ph.D. thesis, University of Exeter.

Hilber, H.M., Hughes, T.J.R., and Taylor, R.L. (1977). Improved numerical dissipation for time integration algorithms in structural dynamics. *Earthquake Engineering & Structural Dynamics* **5**(3), 283–292. doi:10.1002/eqe.4290050306.

Hoger, A. and Carlson, D.E. (1984). Determination of the stretch and rotation in the polar decomposition of the deformation gradient. *Quarterly of Applied Mathematics* **42**, 113–117.

Holmes, P., Lumley, J.L., and Berkooz, G. (1996). *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press.

Holzapfel, G.A. (2000). *Nonlinear Solid Mechanics*. Wiley.

Hughes, T.J.R. (2000). *The Finite Element Method: Linear static and dynamic finite element analysis*. Dover.

Hunt, J.C., Wray, A., and Moin, P. (1988). Eddies, streams, and convergence zones in turbulent flows. In *The Proceedings of the 1988 Summer Program*, 193–208. Center for Turbulense Research, Stanford University. `http://ctr.stanford.edu/Summer/201306111537.pdf`.

Jameson, A. (1968). Solution of the equation $AX + XB = C$ by inversion of an $M \times M$ or $N \times N$ matrix. *SIAM Journal on Applied Mathematics* **16**(5), 1020–1023. doi:10.1137/0116083.

Kang, C.K., Aono, H., Cesnik, C.E.S., and Shyy, W. (2011). Effects of flexibility on the aerodynamic performance of flapping wings. *Journal of Fluid Mechanics* **689**, 32–74. doi:10.1017/jfm.2011.428.

Kim, J. and Moin, P. (1985). Application of a fractional-step method to incompressible navier-stokes equations. *Journal of Computational Physics* **59**(2), 308–323. doi:10.1016/0021-9991(85)90148-2.

Koehler, C., Wischgoll, T., Dong, H., and Gaston, Z. (2011). Vortex visualization in ultra low reynolds number insect flight. *Visualization and Computer Graphics, IEEE Transactions on* **17**(12), 2071–2079. doi:10.1109/TVCG.2011.260.

Kuhl, D. and Crisfield, M.A. (1999). Energy-conserving and decaying algorithms in non-linear structural dynamics. *International Journal for Numerical Methods in Engineering* **45**(5), 569–599. doi:10.1002/(SICI)1097-0207(19990620)45:5<569::AID-NME595>3.0.CO;2-A.

Kuhl, D. and Ramm, E. (1996). Constraint energy momentum algorithm and its application to non-linear dynamics of shells. *Computer methods in applied mechanics and engineering* **136**(3-4), 293–315. doi:10.1016/0045-7825(95)00963-9.

Kuznetsov, Y.A. (2004). *Elements of Applied Bifurcation Theory*, volume 112 of *Applied Mathematical Sciences*. Springer, 3rd edition. doi:10.1007/978-1-4757-3978-7.

Kweon, J. and Choi, H. (2010). Sectional lift coefficient of a flapping wing in hovering motion. *Physics of Fluids* **22**(7), 071703. doi:10.1063/1.3471593.

Lehman, F.O. and Dickinson, M.H. (1997). The changes in power requirements and muscle efficiency during elevated force production in the fruit fly Drosophila melanogaster. *Journal of Experimental Biology* **200**(7), 1133–1143.

Li, X., Demmel, J., Gilbert, J., Grigori, L., Shao, M., and Yamazaki, I. (1999). SuperLU Users' Guide. Technical Report LBNL-44289, Lawrence Berkeley National Laboratory. `http://crd.lbl.gov/~xiaoye/SuperLU/`. Last update: October 2011.

Li, X.S. (2005). An overview of SuperLU: Algorithms, implementation, and user interface. *ACM Transactions on Mathematical Software* **31**(3), 302–325. doi:10.1145/1089014.1089017.

MacNeice, P., Olson, K.M., Mobarry, C., de Fainchtein, R., and Packer, C. (2000). PARAMESH: A parallel adaptive mesh refinement community toolkit. *Computer Physics Communications* **126**(3), 330–354. doi:10.1016/S0010-4655(99)00501-9.

Meirovitch, L. (2001). *Fundamentals of Vibrations*. McGraw-Hill.

Miller, L.A. and Peskin, C.S. (2005). A computational fluid dynamics of 'clap and fling' in the smallest insects. *Journal of Experimental Biology* **208**, 195–212. doi:10.1242/jeb.01376.

Mountcastle, A.M. and Daniel, T.L. (2009). Aerodynamic and functional consequences of wing compliance. *Experiments in Fluids* **46**(5), 873–882. doi:10.1007/s00348-008-0607-0.

Nayfeh, A.H. and Balachandran, B. (1995). *Applied Nonlinear Dynamics: Analytical, Computational, and Experimental Methods*. Wiley.

Newmark, N.M. (1959). A method of computation for structural dynamics. *Journal of the Engineering Mechanics Division ASCE* **85**(EM 3), 67–94.

Norris, A.N. (2007). Invariants of $C^{1/2}$ in terms of the invariants of C. *Journal of Mechanics of Materials and Structures* **2**(9), 1805–1812. doi:10.2140/jomms.2007.2.1805.

O'Donnell, B.J. and Helenbrook, B.T. (2007). Proper orthogonal decomposition and incompressible flow: An application to particle modeling. *Computers and Fluids* **36**(7), 1174–1186. doi:10.1016/j.compfluid.2006.12.004.

Ogden, R.W. (1984). *Non-linear Elastic Deformations*. Wiley.

O'Hara, R.P. and Palazotto, A.N. (2012). The morphological characterization of the forewing of the manduca sexta species for the application of biomimetic flapping wing micro air vehicles. *Bioinspiration & Biomimetics* **7**(4), 046011. doi:10.1088/1748-3182/7/4/046011.

Pai, F. (2007). *Highly Flexible Structures: Modeling, Computation, and Experimentation*. AIAA. doi:10.2514/4.861925.

Pai, P.F., Chernova, D.K., and Palazotto, A.N. (2009). Nonlinear modeling and vibration characterization of mav flapping wings. In *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. Palm Springs, California. doi:10.2514/6.2009-2415.

Pai, P.F. and Nayfeh, A.H. (1994). A new method for the modeling of geometric nonlinearities in structures. *Computers & Structures* **53**(4), 877–895. doi:10.1016/0045-7949(94)90376-X.

Pai, P.F. and Palazotto, A.N. (1995). Polar decomposition theory in nonlinear analyses of solids and structures. *Journal of Engineering Mechanics* **121**(4), 568–581. doi:10.1061/(ASCE)0733-9399(1995)121:4(568).

Preidikman, S. (1998). *Numerical Simulations of Interactions Amoung Aerodynamics, Structural Dynamics, and Control Systems*. Ph.D. thesis, Virginia Polytechnic Institute and State University.

Preidikman, S. and Mook, D.T. (1998). On the development of a passive-damping system for wind-excited oscillations of long-span bridges. *Journal of Wind Engineering and Industrial Aerodynamics* **77-78**, 443–456. doi:10.1016/S0167-6105(98)00163-9.

Ramamurti, R. and Sandberg, W.C. (2002). A three-dimensional computational study of the aerodynamic mechanisms of insect flight. *Journal of Experimental Biology* **205**(10), 1507–1518.

Rosati, L. (1999). Derivatives and rates of the stretch and rotation tensors. *Journal of elasticity* **56**(3), 213–230. doi:10.1023/A:1007663620943.

Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition. doi:10.1137/1.9780898718003.

Sane, S.P. and Dickinson, M.H. (2001). The control of flight force by a flapping wing: Lift and drag production. *Journal of Experimental Biology* **204**, 2607–2626.

Schenato, L. (2003). *Analysis and Control of Flapping Flight: from Biological to Robotic Insects*. Ph.D. thesis, University of California at Berkeley.

Shampine, L.F. and Reichelt, M.W. (1997). The MATLAB ode suite. *SIAM journal on scientific computing* **18**(1), 1–22. doi:10.1137/S1064827594276424.

Shyy, W., Aono, H., Chimakurthi, S., Trizila, P., Kang, C.K., Cesnik, C., and Liu, H. (2010). Recent progress in flapping wing aerodynamics and aeroelasticity. *Progress in Aerospace Sciences* **46**(7), 284–327. doi:10.1016/j.paerosci.2010.01.001.

Sims, T.W. (2010). *A Structural Dynamic Analysis of a Manduca Sexta Forewing*. Ph.D. thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH.

Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. Part I: Coherent structures. *Quarterly of Applied Mathematics* **45**, 561–571.

Smith, D.R. (1993). *An Introduction to Continuum Mechanics - after Truesdell and Noll*, volume 22 of *Solid Mechanics and Its Applications*. Kluwer Academic Publishers.

Song, D., Wang, H., Zeng, L., and Yin, C. (2001). Measuring the camber deformation of a dragonfly wing using projected comb fringe. *Review of Scientific Instruments* **72**(5), 2450–2454. doi:10.1063/1.1364664.

Song, F., Xiao, K., Bai, K., and Bai, Y. (2007). Microstructure and nanomechanical properties of the wing membrane of dragonfly. *Materials Science and Engineering: A* **457**(1-2), 254–260. doi:10.1016/j.msea.2007.01.136.

Spedding, G.R., Rosén, M., and Hedenström, A. (2003). A family of vortex wakes generated by a thrush nightingale in free flight in a wind tunnel over its entire natural range of flight speeds. *Journal of Experimental Biology* **206**, 2313–2344. doi:10.1242/jeb.00423.

Stanford, B., Beran, P., Snyder, R., and Patil, M. (2013). Stability and power optimality in time-periodic flapping wing structures. *Journal of Fluids and Structures* **38**, 238–254. doi:10.1016/j.jfluidstructs.2012.12.006.

Stroud, A.H. (1971). *Approximate calculations of multiple integrals*. Prentice-Hall.

Sun, M. and Tang, J. (2002a). Lift and power requirements of hovering flight in Drosophila virilis. *Journal of Experimental Biology* **205**(16), 2413–2427.

Sun, M. and Tang, J. (2002b). Unsteady aerodynamic force generation by a model fruit fly wing in flapping motion. *Journal of Experimental Biology* **205**(1), 55–70.

Valdez, M. (2008). Unsteady vortex lattice method code. Personal Communication.

Valdez, M., Preidikman, S., and Massa, J.C. (2006). Aerodinámica de flujos bidemensionales e. inestacionaries dominados por vorticidad. *Mecánica Computacional* **25**, 2333–2357. In Spanish.

Vanella, M. (2010). *A fluid structure interaction strategy with application to low Reynolds number flapping flight*. Ph.D. thesis, University of Maryland, College Park. `http://hdl.handle.net/1903/10830`.

Vanella, M., Fitzgerald, T., Preidikman, S., Balaras, E., and Balachandran, B. (2009). Influence of flexibility on the aerodynamic performance of a hovering wing. *Journal of Experimental Biology* **212**(1), 95–105. doi:10.1242/jeb.016428.

Walker, S.M., Thomas, A.L.R., and Taylor, G.K. (2009a). Deformable wing kinematics in free-flying hoverflies. *Journal of the Royal Society Interface* doi:10.1098/rsif.2009.0120.

Walker, S.M., Thomas, A.L.R., and Taylor, G.K. (2009b). Deformable wing kinematics in the desert locust: how and why do camber, twist and topography vary through the stroke? *Journal of the Royal Society Interface* **6**(38), 735–747. doi:10.1098/rsif.2008.0435.

Walker, S.M., Thomas, A.L.R., and Taylor, G.K. (2009c). Photogrammetric reconstruction of high-resolution surface topographies and deformable wing kinematics of tethered locusts and free-flying hoverflies. *Journal of the Royal Society Interface* **6**(33), 351–366. doi:10.1098/rsif.2008.0245.

Wang, J.K. and Sun, M. (2005). A computational study of the aerodynamics and forewing-hindwing interaction of a model dragonfly in forward flight. *Journal of Experimental Biology* **208**(19), 3785–3804. doi:10.1242/jeb.01852.

Wang, Z.J. (2000a). Two dimensional mechanism for insect hovering. *Physical Review Letters* **85**(10), 2216–2219. doi:10.1103/PhysRevLett.85.2216.

Wang, Z.J. (2000b). Vortex shedding and frequency selection in flapping flight. *Journal of Fluid Mechanics* **410**, 323–341. doi:10.1017/S0022112099008071.

Wang, Z.J. (2005). Dissecting insect flight. *Annual Review of Fluid Mechanics* **37**(1), 183–210. doi:10.1146/annurev.fluid.36.050802.121940.

Wang, Z.J., Birch, J.M., and Dickinson, M.H. (2004). Unsteady forces and flows in low reynolds number hovering flight: two-dimensional computations vs robotic wing experiments. *Journal of Experimental Biology* **207**, 449–460. doi:10.1242/jeb.00739.

Weis-Fogh, T. (1973). Quick estimates of flight fitness in hovering animals, including novel mechanics for lift production. *Journal of Experimental Biology* **59**(1), 169–230.

Willmott, A.P. and Ellington, C.P. (1997). The mechanics of flight in the hawkmoth manduca sexta. *Journal of Experimental Biology* **200**, 2705–2722.

Wood, W.L., Bossak, M., and Zienkiewicz, O.C. (1980). An alpha modification of Newmark's method. *International Journal for Numerical Methods in Engineering* **15**(10), 1562–1566. doi:10.1002/nme.1620151011.

Wootton, R.J. (1992). Functional morphology of insect wings. *Annual Review of Entomology* **37**, 113–140. doi:10.1146/annurev.en.37.010192.000553.

Wootton, R.J. (1999). Invertebrate paraxial locomotory appendages: design, deformation, and control. *Journal of Experimental Biology* **202**, 3333–3345.

Wootton, R.J., Herbert, R.C., Young, P.G., and Evans, K.E. (2003). Approaches to the structural modelling of insect wings. *Philosophical Transactions of the Royal Society of London* **358**(1437), 1577–1587. doi:10.1098/rstb.2003.1351.

Yang, J., Preidikman, S., and Balaras, E. (2008). A strongly coupled, embedded-boundary method for fluid-structure interactions of elastically mounted rigid bodies. *Journal of Fluids and Structures* **24**(2), 167–182. doi:10.1016/j.jfluidstructs.2007.08.002.

Zhao, L., Huang, Q., Deng, X., and Sane, S.P. (2010). Aerodynamic effects of flexibility in flapping wings. *Journal of the Royal Society Interface* **7**(44), 485–497. doi:10.1098/rsif.2009.0200.